(19) EP Office europeen des brevets

(11) EP 1 022 641 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
26.07.2000 Bulletin 2000/30

(21) Application number: **00100018.1**

(22) Date of filing: **16.04.1991**

(51) Int. Cl.[7]: **G06F 1/04**, G06F 1/12, G06F 13/00, G06F 13/16, G06F 13/36, G06F 13/38, G11C 7/00, G11C 8/00, G11C 8/04

(84) Designated Contracting States:
DE FR GB IT

(30) Priority: **18.04.1990 US 510898**

(62) Document number(s) of the earlier application(s) in accordance with Art. 76 EPC:
91908374.1 / 0 525 068

(71) Applicant: **RAMBUS INC.**
Mountain View, CA 94040 (US)

(72) Inventors:
- **Farmwald, Michael**
  Portola Valley, California 94028 (US)
- **Horowitz, Mark**
  Palo Alto, California 94306 (US)

(74) Representative:
Eisenführ, Speiser & Partner
Martinistrasse 24
28195 Bremen (DE)

Remarks:
This application was filed on 04 - 01 - 2000 as a divisional application to the application mentioned under INID code 62.

(54) Integrated circuit I/o using a high performance bus interface

(57) The present invention includes a memory subsystem comprising at least two semiconductor devices (15, 16, 17), including at least one memory device (15, 16 or 17), connected to a bus (18), where the bus includes a plurality of bus lines for carrying substantially all address, data and control information needed by said memory devices (15, 16 or 17), where the control information includes device-select information and the bus (18) has substantially fewer bus lines than the number of bits in a single address, and the bus (18) carries device-select information without the need for separated device-select lines connected directly to individual devices. The present invention also includes a protocol for master and slave devices to communicate on the bus (18) and for registers in each device to differentiate each device and allow bus requests to be directed to a single or to all devices (15, 16, 17). The present invention includes modifications to prior-art devices to allow them to implement the new features of this invention. In a preferred implementation, 8 bus data lines and an Address Valid bus line carry address, data and control information for memory addresses up to 40 bits wide.

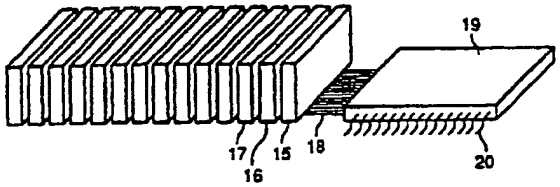

FIG. 3

EP 1 022 641 A1

Description

FIELD OF THE INVENTION

[0001] An integrated circuit bus interface for computer and video systems is described which allows high speed transfer of blocks of data, particularly to and from memory devices, with reduced power consumption and increased system reliability. A new method of physically implementing the bus architecture is also described.

BACKGROUND OF THE INVENTION

[0002] Semiconductor computer memories have traditionally been designed and structured to use one memory device for each bit, or small group of bits, of any individual computer word, where the word size is governed by the choice of computer. Typical word sizes range from 4 to 64 bits. Each memory device typically is connected in parallel to a series of address lines and connected to one of a series of data lines. When the computer seeks to read from or write to a specific memory location, an address is put on the address lines and some or all of the memory devices are activated using a separate device select line for each needed device. One or more devices may be connected to each data line but typically only a small number of data lines are connected to a single memory device. Thus data line 0 is connected to device(s) 0, data line 1 is connected to device(s) 1, and so on. Data is thus accessed or provided in parallel for each memory read or write operation. For the system to operate properly, every single memory bit in every memory device must operate dependably and correctly.

[0003] To understand the concept of the present invention, it is helpful to review the architecture of conventional memory devices. Internal to nearly all types of memory devices (including the most widely used Dynamic Random Access Memory (DRAM), Static RAM (SRAM) and Read Only Memory (ROM) devices), a large number of bits are accessed in parallel each time the system carries out a memory access cycle. However, only a small percentage of accessed bits which are available internally each time the memory device is cycled ever make it across the device boundary to the external world.

[0004] Referring to Fig. 1, all modern DRAM, SRAM and ROM designs have internal architectures with row (word) lines 5 and column (bit) lines 6 to allow the memory cells to tile a two dimensional area 1. One bit of data is stored at the intersection of each word and bit line. When a particular word line is enabled, all of the corresponding data bits are transferred onto the bit lines. Some prior art DRAMs take advantage of this organization to reduce the number of pins needed to transmit the address. The address of a given memory cell is split into two addresses, row and column, each of which can be multiplexed over a bus only half as wide as the memory cell address of the prior art would have required.

COMPARISON WITH PRIOR ART

[0005] Prior art memory systems have attempted to solve the problem of high speed access to memory with limited success. U.S. Patent No. 3,821,715 (Hoff et. al.), was issued to Intel Corporation for the earliest 4-bit micro-processor. That patent describes a bus connecting a single central processing unit (CPU) with multiple RAMs and ROMs. That bus multiplexes addresses and data over a 4-bit wide bus and uses point-to-point control signals to select particular RAMs or ROMs. The access time is fixed and only a single processing element is permitted. There is no block-mode type of operation, and most important, not all of the interface signals between the devices are based (the ROM and RAM control lines and the RAM select lines are point-to-point).

[0006] In U.S. Patent No. 4,315,308 (Jackson), a bus connecting a single CPU to a bus interface unit is described. The invention uses multiplexed address, data, and control information over a single 16-bit wide bus. Block-mode operations are defined, with the length of the block sent as part of the control sequence. In addition, variable access-time operations using a "stretch" cycle signal are provided. There are no multiple processing elements and no capability for multiple outstanding requests, and again, not all of the interface signals are bused.

[0007] In U.S. Patent No. 4,449,207 (Kung, et. al.), a DRAM is described which multiplexes address and data on an internal bus. The external interface to this DRAM is conventional, with separate control, address and data connections.

[0008] In U.S. Patent Nos. 4,764,846 and 4,706,166 (Go), a 3-D package arrangement of stacked die with connections along a single edge is described. Such packages are difficult to use because of the point-to-point wiring required to interconnect conventional memory devices with processing elements. Both patents describe complex schemes for solving these problems. No attempt is made to solve the problem by changing the interface.

[0009] In U.S. Patent No. 3,969,706 (Proebsting, et. al.), the current state-of-the-art DRAM interface is described. The address is two-way multiplexed, and there are separate pins for data and control (RAS, CAS, WE, CS). The number of pins grows with the size of the DRAM, and many of the connections must be made point-to-point in a memory system using such DRAMs.

[0010] There are many backplane buses described in the prior art, but not in the combination described or having the features of this invention. Many backplane buses multiplex addresses and data on a single bus (e.g., the NU bus). ELXSI and others have implemented split-transaction buses (U.S. Patent No. 4,595,923 and 4,481,625 (Roberts)). ELXSI has also implemented a relatively low-voltage-swing current-mode ECL driver (approximately 1 V swing). Address-space registers are implemented on most backplane buses, as is some form of block mode operation.

[0011] Nearly all modern backplane buses implement some type of arbitration scheme, but the arbitration scheme used in this invention differs from each of these. U.S. Patent Nos. 4,837,682 (Culler), 4,818,985 (Ikeda), 4,779,089 (Theus) and 4,745,548 (Blahut) describe prior art schemes. All involve either log N extra signals, (Theus, Blahut), where N is the number of potential bus requestors, or additional delay to get control of the bus (Ikeda, Culler). None of the buses described in patents or other literature use only bused connections. All contain some point-to-point connections on the backplane. None of the other aspects of this invention such as power reduction by fetching each data block from a single device or compact and low-cost 3-D packaging even apply to backplane buses.

[0012] The clocking scheme used in this invention has not been used before and in fact would be difficult to implement in backplane buses due to the signal degradation caused by connector stubs. U.S. Patent No. 4,247,817 (Heller) describes a clocking scheme using two clock lines, but relies on ramp-shaped clock signals in contrast to the normal rise-time signals used in the present invention.

[0013] In U.S. Patent No. 4,646,279 (Voss), a video RAM is described which implements a parallel-load, serial-out shift register on the output of a DRAM. This generally allows greatly improved bandwidth (and has been extended to 2, 4 and greater width shift-out paths.) The rest of the interfaces to the DRAM (RAS, CAS, multiplexed address, etc.) remain the same as for conventional DRAMS.

[0014] One object of the present invention is to use a new bus interface built into semiconductor devices to support high-speed access to large blocks of data from a single memory device by an external user of the data, such as a microprocessor, in an efficient and cost-effective manner.

[0015] Another object of this invention is to provide a clocking scheme to permit high speed clock signals to be sent along the bus with minimal clock skew between devices.

[0016] Another object of this invention is to allow mapping out defective memory devices or portions of memory devices.

[0017] Another object of this invention is to provide a method for distinguishing otherwise identical devices by assigning a unique identifier to each device.

[0018] Yet another object of this invention is to provide a method for transferring address, data and control information over a relatively narrow bus and to provide a method of bus arbitration when multiple devices seek to use the bus simultaneously.

[0019] Another object of this invention is to provide a method of distributing a high-speed memory cache within the DRAM chips of a memory system which is much more effective than previous cache methods.

[0020] Another object of this invention is to provide devices, especially DRAMa, suitable for use with the bus architecture of the invention.

SUMMARY OF INVENTION

[0021] The present invention includes a memory subsystem comprising at least two semiconductor devices, including at least one memory device, connected in parallel to a bus, where the bus includes a plurality of bus lines for carrying substantially all address, data and control information needed by said memory devices, where the control information includes device-select information and the bus has substantially fewer bus lines than the number of bits in a single address, and the bus carries device-select information without the need for separate device-select lines connected directly to individual devices.

[0022] Referring to Fig. 2, a standard DRAM **13**, **14**, ROM (or SRAM) **12**, microprocessor CPU **11**, I/O device, disk controller or other special purpose device such as a high speed switch is modified to use a wholly bus-based interface rather than the prior art combination of point-to-point and bus-based wiring used with conventional versions of these devices. The new bus includes clock signals, power and multiplexed address, data and control signals. In a preferred implementation, 8 bus data lines and an AddressValid bus line carry address, data and control information for memory addresses up to 40 bits wide. Persons skilled in the art will recognize that 16 bus data lines or other numbers of bus data lines can be used to implement the teaching of this invention. The new bus is used to connect elements such as memory, peripheral, switch and processing units.

[0023] In the system of this invention, DRAMs and other devices receive address and control information over the bus and transmit or receive requested data over the same bus. Each memory device contains only a single bus interface with no other signal pins. Other devices that may be included in the system can connect to the bus and other non-bus lines, such as input/output lines. The bus supports large data block transfers and split transactions to allow a user to achieve high bus utilization. This ability to rapidly read or write a large block of data to one single device at a time is

an important advantage of this invention.

[0024] The DRAMs that connect to this bus differ from conventional DRAMs in a number of ways. Registers are provided which may store control information, device identification, device-type and other information appropriate for the chip such as the address range for each independent portion of the device. New bus interface circuits must be added and the internals of prior art DRAM devices need to be modified so they can provide and accept data to and from the bus at the peak data rate of the bus. This requires changes to the column access circuitry in the DRAM, with only a minimal increase in die size. A circuit is provided to generate a low skew internal device clock for devices on the bus, and other circuits provide for demultiplexing input and multiplexing output signals.

[0025] High bus bandwidth is achieved by running the bus at a very high clock rate (hundreds of MHz). This high clock rate is made possible by the constrained environment of the bus. The bus lines are controlled-impedance, doubly-terminated lines. For a data rate of 500 MHz, the maximum bus propagation time is less than 1 ns (the physical bus length is about 10 cm). In addition, because of the packaging used, the pitch of the pins can be very close to the pitch of the pads. The loading on the bus resulting from the individual devices is very small. In a preferred implementation, this generally allows stub capacitances of 1-2 pF and inductances of 0.5 - 2 nH. Each device **15, 16, 17**, shown in Figure 3, only has pins on one side and these pins connect directly to the bus **18**. A transceiver device **19** can be included to interface multiple units to a higher order bus through pins **20**.

[0026] A primary result of the architecture of this invention is to increase the bandwidth of DRAM access. The invention also reduces manufacturing and production costs, power consumption, and increases packing density and system reliability.

BRIEF DESCRIPTION OF TEE DRAWINGS

[0027]

Figure 1 is a diagram which illustrates the basic 2-D organization of memory devices.
Figure 2 is a schematic block diagram which illustrates the parallel connection of all bus lines and the serial Reset line to each device in the system.
Figure 3 is a perspective view of a system of the invention which illustrates the 3-D packaging of semiconductor devices on the primary bus.
Figure 4 shows the format of a request packet.
Figure 5 shows the format of a retry response from a slave.
Figure 6 shows the bus cycles after a request packet collision occurs on the bus and how arbitration is handled.
Figure 7 shows the timing whereby signals from two devices can overlap temporarily and drive the bus at the same time.
Figure 8 shows the connection and timing between bus clocks and devices on the bus.
Figure 9 is a perspective view showing how transceivers can be used to connect a number of bus units to a transceiver bus. Figure 10 is a block and schematic diagram of input/output circuitry used to connect devices to the bus.
Figure 11 is a schematic diagram of a clocked sense-amplifier used as a bus input receiver.
Figure 12 is a block diagram showing how the internal device clock is generated from two bus clock signals using a set of adjustable delay lines.
Figure 13 is a timing diagram showing the relationship of signals in the block diagram of Figure 12.
Figure 14 is timing diagram of a preferred means of implementing the reset procedure of this invention.
Figure 15 is a diagram illustrating the general organization of a 4 Mbit DRAM divided into 8 subarrays.

DETAILED DESCRIPTION

[0028] The present invention is designed to provide a high speed, multiplexed bus for communication between processing devices and memory devices and to provide devices adapted for use in the bus system. The invention can also be used to connect processing devices and other devices, such as I/O interfaces or disk controllers, with or without memory devices on the bus. The bus consists of a relatively small number of lines connected in parallel to each device on the bus. The bus carries substantially all address, data and control information needed by devices for communication with other devices on the bus. In many systems using the present invention, the bus carries almost every signal between every device in the entire system. There is no need for separate device-select lines since device-select information for each device on the bus is carried over the bus. There is no need for separate address and data lines because address and data information can be sent over the same lines. Using the organization described herein, very large addresses (40 bits in the preferred implementation) and large data blocks (1024 bytes) can be sent over a small number of bus lines (8 plus one control line in the preferred implementation).

[0029] Virtually all of the signals needed by a computer system can be sent over the bus. Persons skilled in the art

recognize that certain devices, such as CPUs, may be connected to other signal lines and possibly to independent buses, for example a bus to an independent cache memory, in addition to the bus of this invention. Certain devices, for example cross-point switches, could be connected to multiple, independent buses of this invention. In the preferred implementation, memory devices are provided that have no connections other than the bus connections described herein and CPUs are provided that use the bus of this invention as the principal, if not exclusive, connection to memory and to other devices on the bus.

[0030] All modern DRAM, SRAM and ROM designs have internal architectures with row (word) and column (bit) lines to efficiently tile a 2-D area. Referring to Fig. 1, one bit of data is stored at the intersection of each word line 5 and bit line 6. When a particular word line is enabled, all of the corresponding data bits are transferred onto the bit lines. This data, about 4000 bits at a time in a 4 MBit DRAM, is then loaded into column sense amplifiers 3 and held for use by the I/O circuits.

[0031] In the invention presented here, the data from the sense amplifiers is enabled 32 bits at a time onto an internal device bus running at approximately 125 MHz. This internal device bus moves the data to the periphery of the devices where the data is multiplexed into an 8-bit wide external bus interface, running at approximately 500 MHz.

[0032] The bus architecture of this invention connects master or bus controller devices, such as CPUs, Direct Memory Access devices (DMAs) or Floating Point Units (FPUs), and slave devices, such as DRAM, SPAM or ROM memory devices. A slave device responds to control signals; a master sends control signals. Persons skilled in the art realize that some devices may behave as both master and slave at various times, depending on the mode of operation and the state of the system. For example, a memory device will typically have only slave functions, while a DMA controller, disk controller or CPU may include both slave and master functions. Many other semiconductor devices, including I/O devices, disk controllers, or other special purpose devices such as high speed switches can be modified for use with the bus of this invention.

[0033] Each semiconductor device contains a set of internal registers, preferably including a device identification (device ID) register, a device-type descriptor register, control registers and other registers containing other information relevant to that type of device. In a preferred implementation, semiconductor devices connected to the bus contain registers which specify the memory addresses contained within that device and access-time registers which store a set of one or more delay times at which the device can or should be available to send or receive data.

[0034] Most of these registers can be modified and preferably are set as part of an initialization sequence that occurs when the system is powered up or reset. During the initialization sequence each device on the bus is assigned a unique device ID number, which is stored in the device ID register. A bus master can then use these device ID numbers to access and set appropriate registers in other devices, including access-time registers, control registers, and memory registers, to configure the system. Each slave may have one or several access-time registers (four in a preferred embodiment). In a preferred embodiment, one access-time register in each slave is permanently or semi-permanently programmed with a fixed value to facilitate certain control functions. A preferred implementation of an initialization sequence is described below in more detail.

[0035] All information sent between master devices and slave devices is sent over the external bus, which, for example, may be 8 bits wide. This is accomplished by defining a protocol whereby a master device, such as a microprocessor, seizes exclusive control of the external bus (i.e., becomes the bus master) and initiates a bus transaction by sending a request packet (a sequence of bytes comprising address and control information) to one or more slave devices on the bus. An address can consist of 16 to 40 or more bits according to the teachings of this invention. Each slave on the bus must decode the request packet to see if that slave needs to respond to the packet. The slave that the packet is directed to must then begin any internal processes needed to carry out the requested bus transaction at the requested time. The requesting master may also need to transact certain internal processes before the bus transaction begins. After a specified access time the slave(s) respond by returning one or more bytes (8 bits) of data or by storing information made available from the bus. More than one access time can be provided to allow different types of responses to occur at different times.

[0036] A request packet and the corresponding bus access are separated by a selected number of bus cycles, allowing the bus to be used in the intervening bus cycles by the same or other masters for additional requests or brief bus accesses. Thus multiple, independent accesses are permitted, allowing maximum utilization of the bus for transfer of short blocks of data. Transfers of long blocks of data use the bus efficiently even without overlap because the overhead due to bus address, control and access times is small compared to the total time to request and transfer the block.

Device Address Mapping

[0037] Another unique aspect of this invention is that each memory device is a complete, independent memory subsystem with all the functionality of a prior art memory board in a conventional backplane-bus computer system. Individual memory devices may contain a single memory section or may be subdivided into more than one discrete memory section. Memory devices preferably include memory address registers for each discrete memory section. A failed mem-

ory device (or even a subsection of a device) can be "mapped out" with only the loss of a small fraction of the memory, maintaining essentially full system capability. Mapping out bad devices can be accomplished in two ways, both compatible with this invention.

[0038] The preferred method uses address registers in each memory device (or independent discrete portion thereof) to store information which defines the range of bus addresses to which this memory device will respond. This is similar to prior art schemes used in memory boards in conventional backplane bus systems. The address registers can include a single pointer, usually pointing to a block of known size, a pointer and a fixed or variable block size value or two pointers, one pointing to the beginning and one to the end (or to the "top" and "bottom") of each memory block. By appropriate settings of the address registers, a series of functional memory devices or discrete memory sections can be made to respond to a contiguous range of addresses, giving the system access to a contiguous block of good memory, limited primarily by the number of good devices connected to the bus. A block of memory in a first memory device or memory section can be assigned a certain range of addresses, then a block of memory in a next memory device or memory section can be assigned addresses starting with an address one higher (or lower, depending on the memory structure) than the last address of the previous block.

[0039] Preferred devices for use in this invention include device-type register information specifying the type of chip, including how much memory is available in what configuration on that device. A master can perform an appropriate memory test, such as reading and writing each memory cell in one or more selected orders, to test proper functioning of each accessible discrete portion of memory (based in part on information like device ID number and device-type) and write address values (up to 40 bits in the preferred embodiment, 10^{12} bytes), preferably contiguous, into device address-space registers. Non-functional or impaired memory sections can be assigned a special address value which the system can interpret to avoid using that memory.

[0040] The second approach puts the burden of avoiding the bad devices on the system master or masters. CPUs and DMA controllers typically have some sort of translation look-aside buffers (TLBs) which map virtual to physical (bus) addresses. With relatively simple software, the TLBs can be programmed to use only working memory (data structures describing functional memories are easily generated). For masters which don't contain TLBs (for example, a video display generator), a small, simple RAM can be used to map a contiguous range of addresses onto the addresses of the functional memory devices.

[0041] Either scheme works and permits a system to have a significant percentage of non-functional devices and still continue to operate with the memory which remains. This means that systems built with this invention will have much improved reliability over existing systems, including the ability to build systems with almost no field failures.

Bus

[0042] The preferred bus architecture of this invention comprises 11 signals: BusData[0:7]; AddrValid; Clk1 and Clk2; plus an input reference level and power and ground lines connected in parallel to each device. Signals are driven onto the bus during conventional bus cycles. The notation "Signal[i:j]" refers to a specific range of signals or lines, for example, BusData[0:7] means BusData0, BusData1, . . ., BusData7. The bus lines for BusData[0:7] signals form a byte-wide, multiplexed data/address/control bus. AddrValid is used to indicate when the bus is holding a valid address request, and instructs a slave to decode the bus data as an address and, if the address is included on that slave, to handle the pending request. The two clocks together provide a synchronized, high speed clock for all the devices on the bus. In addition to the bused signals, there is one other line (ResetIn, ResetOut) connecting each device in series for use during initialization to assign every device in the system a unique device ID number (described below in detail).

[0043] To facilitate the extremely high data rate of this external bus relative to the gate delays of the internal logic, the bus cycles are grouped into pairs of even/odd cycles. Note that all devices connected to a bus should preferably use the same even/odd labeling of bus cycles and preferably should begin operations on even cycles. This is enforced by the clocking scheme.

Protocol and Bus Operation

[0044] The bus uses a relatively simple, synchronous, split-transaction, block-oriented protocol for bus transactions. One of the goals of the system is to keep the intelligence concentrated in the masters, thus keeping the slaves as simple as possible (since there are typically many more slaves than masters). To reduce the complexity of the slaves, a slave should preferably respond to a request in a specified time, sufficient to allow the slave to begin or possibly complete a device-internal phase including any internal actions that must precede the subsequent bus access phase. The time for this bus access phase is known to all devices on the bus - each master being responsible for making sure that the bus will be free when the bus access begins. Thus the slaves never worry about arbitrating for the bus. This approach eliminates arbitration in single master systems, and also makes the slave-bus interface simpler.

[0045] In a preferred implementation of the invention, to initiate a bus transfer over the bus, a master sends out a

request packet, a contiguous series of bytes containing address and control information. It is preferable to use a request packet containing an even number of bytes and also preferable to start each packet on an even bus cycle.

[0046] The device-select function is handled using the bus data lines. AddrValid is driven, which instructs all slaves to decode the request packet address, determine whether they contain the requested address, and if they do, provide the data back to the master (in the case of a read request) or accept data from the master (in the case of a write request) in a data block transfer. A master can also select a specific device by transmitting a device ID number in a request packet. In a preferred implementation, a special device ID number is chosen to indicate that the packet should be interpreted by all devices on the bus. This allows a master to broadcast a message, for example to set a selected control register of all devices with the same value.

[0047] The data block transfer occurs later at a time specified in the request packet control information, preferably beginning on an even cycle. A device begins a data block transfer almost immediately with a device-internal phase as the device initiates certain functions, such as setting up memory addressing, before the bus access phase begins. The time after which a data block is driven onto the bus lines is selected from values stored in slave access-time registers. The timing of data for reads and writes is preferably the same; the only difference is which device drives the bus. For reads, the slave drives the bus and the master latches the values from the bus. For writes the master drives the bus and the selected slave latches the values from the bus.

[0048] In a preferred implementation of this invention shown in Figure 4, a request packet **22** contains 6 bytes of data -- 4.5 address bytes and 1.5 control bytes. Each request packet uses all nine bits of the multiplexed data/address lines (AddrValid **23** + BusData[0:7] **24**) for all six bytes of the request packet. Setting **23** AddrValid = **1** in an otherwise unused even cycle indicates the start of an request packet (control information). In a valid request packet, AddrValid **27** must be 0 in the last byte. Asserting this signal in the last byte invalidates the request packet. This is used for the collision detection and arbitration logic (described below). Bytes **25-26** contain the first **35** address bits, Address[0:35]. The last byte contains AddrValid **27** (the invalidation switch) and **28**, the remaining address bits, Address[36:39], and BlockSize[0:3] (control information).

[0049] The first byte contains two 4 bit fields containing control information, AccessType[0:3], an op code (operation code) which, for example, specifies the type of access, and Master[0:3], a position reserved for the master sending the packet to include its master ID number. Only master numbers 1 through 15 are allowed - master number 0 is reserved for special system commands. Any packet with Master[0:3] = 0 is an invalid or special packet and is treated accordingly.

[0050] The AccessType field specifies whether the requested operation is a read or write and the type of access, for example, whether it is to the control registers or other parts of the device, such as memory. In a preferred implementation, AccessType[0] is a Read/Write switch: if it is a 1, then the operation calls for a read from the slave (the slave to read the requested memory block and drive the memory contents onto the bus); if it is a 0, the operation calls for a write into the slave (the slave to read data from the bus and write it to memory). AccessType[1:3] provides up to 8 different access types for a slave. AccessType[1:2] preferably indicates the timing of the response, which is stored in an access-time register, AccessRegN. The choice of access-time register can be selected directly by having a certain op code select that register, or indirectly by having a slave respond to selected op codes with pre-selected access times (see table below). The remaining bit, AccessType[3] may be used to send additional information about the request to the slaves.

[0051] One special type of access is control register access, which involves addressing a selected register in a selected slave. In the preferred implementation of this invention, AccessType[1:3] equal to zero indicates a control register request and the address field of the packet indicates the desired control register. For example, the most significant two bytes can be the device ID number (specifying which slave is being addressed) and the least significant three bytes can specify a register address and may also represent or include data to be loaded into that control register. Control register accesses are used to initialize the access-time registers, so it is preferable to use a fixed response time which can be preprogrammed or even hard wired, for example the value in AccessReg0, preferably 8 cycles. Control register access can also be used to initialize or modify other registers, including address registers.

[0052] The method of this invention provides for access mode control specifically for the DRAMs. One such access mode determines whether the access is page mode or normal RAS access. In normal mode (in conventional DRAMS and in this invention), the DRAM column sense amps or latches have been precharged to a value intermediate between logical 0 and 1. This precharging allows access to a row in the RAM to begin as soon as the access request for either inputs (writes) or outputs (reads) is received and allows the column sense amps to sense data quickly. In page mode (both conventional and in this invention), the DRAM holds the data in the column sense amps or latches from the previous read or write operation. If a subsequent request to access data is directed to the same row, the DRAM does not need to wait for the data to be sensed (it has been sensed already) and access time for this data is much shorter than the normal access time. Page mode generally allows much faster access to data but to a smaller block of data (equal to the number of sense amps). However, if the requested data is not in the selected row, the access time is longer than the normal access time, since the request must wait for the RAM to precharge before the normal mode access can start. Two access-time registers in each DRAM preferably contain the access times to be used for normal and for page-

mode accesses, respectively.

[0053] The access mode also determines whether the DRAM should precharge the sense amplifiers or should save the contents of the sense amps for a subsequent page mode access. Typical settings are "precharge after normal access" and "save after page mode access" but "precharge after page mode access" or "save after normal access" are allowed, selectable modes of operation. The DRAM can also be set to precharge the sense amps if they are not accessed for a selected period of time.

[0054] In page mode, the data stored in the DRAM sense amplifiers may be accessed within much less tine than it takes to read out data in normal mode (~10-20 nS vs. 40-100 nS). This data may be kept available for long periods. However, if these sense amps (and hence bit lines) are not precharged after an access, a subsequent access to a different memory word (row) will suffer a precharge time penalty of about 40-100 nS because the sense amps must precharge before latching in a new value.

[0055] The contents of the sense amps thus may be held and used as a cache, allowing faster, repetitive access to small blocks of data. DRAM-based page-mode caches have been attempted in the prior art using conventional DRAM organizations but they are not very effective because several chips are required per computer word. Such a conventional page-mode cache contains many bits (for example, 32 chips x 4Kbits) but has very few independent storage entries. In other words, at any given point in time the sense amps hold only a few different blocks or memory "locales" (a single block of 4K words, in the example above). Simulations have shown that upwards of 100 blocks are required to achieve high hit rates (>90% of requests find the requested data already in cache memory) regardless of the size of each block. *See, for example*, Anant Agarwal, et. al., "An Analytic Cache Model," *ACM Transactions on Computer Systems*, Vol. 7(2), pp. 184-215 (May 1989).

[0056] The organization of memory in the present invention allows each DRAM to hold one or more (4 for 4MBit DRAMS) separately- addressed and independent blocks of data. A personal computer or workstation with 100 such DRAMs (i.e. 400 blocks or locales) can achieve extremely high, very repeatable hit rates (98-99% on average) as compared to the lower (50-80%), widely varying hit rates using DRAMS organized in the conventional fashion. Further, because of the time penalty associated with the deferred precharge on a "miss" of the page-mode cache, the conventional DRAM-based page-mode cache generally has been found to work less well than no cache at all.

[0057] For DRAM slave access, the access types are preferably used in the following way:

AccessType[1:3]	Use	AccessTime
0	Control Register Access	Fixed, 8[AccessReg0]
1	Unused	Fixed, 8[AccessReg0]
2-3	Unused	AccessReg1
4-5	Page Mode DRAM access	AccessReg2
6-7	Normal DRAM access	AccessReg3

Persons skilled in the art will recognize that a series of available bits could be designated as switches for controlling these access modes. For example:

AccessType[2] = page mode/normal switch
AccessType[3] = precharge/save-data switch

[0058] BlockSize[0:3] specifies the size of the data block transfer. If BlockSize[0] is 0, the remaining bits are the binary representation of the block size (0-7). If BlockSize[0] is 1, then the remaining bits give the block size as a binary power of 2, from 8 to 1024. A zero-length block can be interpreted as a special command, for example, to refresh a DRAM without returning any data, or to change the DRAM from page mode to normal access mode or vice-versa.

BlockSize[0:2]	Number of Bytes in Block
0-7	0-7 respectively
8	8

(continued)

BlockSize[0:2]	Number of Bytes in Block
9	16
10	32
11	64
12	128
13	256
14	512
15	1024

Persons skilled in the art will recognize that other block size encoding schemes or values can be used.

[0059] In most cases, a slave will respond at the selected access time by reading or writing data from or to the bus over bus lines BusData[0:7] and AddrValid will be at logical 0. In a preferred embodiment, substantially each memory access will involve only a single memory device, that is, a single block will be read from or written to a single memory device.

Retry Format

[0060] In some cases, a slave may not be able to respond correctly to a request, e.g., for a read or write. In such a situation, the slave should return an error message, sometimes called a N(o)ACK(nowledge) or retry message. The retry message can include information about the condition requiring a retry, but this increases system requirements for circuitry in both slave and masters. A simple message indicating only that an error has occurred allows for a less complex slave, and the master can take whatever action is needed to understand and correct the cause of the error.

[0061] For example, under certain conditions a slave might not be able to supply the requested data. During a page-mode access, the DRAM selected must be in page mode and the requested address must match the address of the data held in the sense amps or latches. Each DRAM can check for this match during a page-mode access. If no match is found, the DRAM begins precharging and returns a retry message to the master during the first cycle of the data block (the rest of the returned block is ignored). The master then must wait for the precharge time (which is set to accommodate the type of slave in question, stored in a special register, PreChargeReg), and then resend the request as a normal DRAM access (AccessType = 6 or 7).

[0062] In the preferred form of the present invention, a slave signals a retry by driving AddrValid true at the time the slave was supposed to begin reading or writing data. A master which expected to write to that slave must monitor AddrValid during the write and take corrective action if it detects a retry message. Figure 5 illustrates the format of a retry message **28** which is useful for read requests, consisting of **23** AddrValid=1 with Master[0:3] = 0 in the first (even) cycle. Note that AddrValid is normally 0 for data block transfers and that there is no master 0 (only 1 through 15 are allowed). All DRAMs and masters can easily recognize such a packet as an invalid request packet, and therefore a retry message. In this type of bus transaction all of the fields except for Master[0:3] and AddrValid **23** may be used as information fields, although in the implementation described, the contents are undefined. Persons skilled in the art recognize that another method of signifying a retry message is to add a DataInvalid line and signal to the bus. This signal could be asserted in the case of a NACK.

Bus Arbitration

[0063] In the case of a single master, there are by definition no arbitration problems. The master sends request packets and keeps track of periods when the bus will be busy in response to that packet. The master can schedule multiple requests so that the corresponding data block transfers do not overlap.

[0064] The bus architecture of this invention is also useful in configurations with multiple masters. When two or more masters are on the same bus, each master must keep track of all the pending transactions, so each master knows when it can send a request packet and access the corresponding data block transfer. Situations will arise, however, where two or more masters send a request packet at about the same time and the multiple requests must be detected, then sorted out by some sort of bus arbitration.

[0065] There are many ways for each master to keep track of when the bus is and will be busy. A simple method is for each master to maintain a bus-busy data structure, for example by maintaining two pointers, one to indicate the earliest point in the future when the bus will be busy and the other to indicate the earliest point in the future when the bus

will be free, that is, the end of the latest pending data block transfer. Using this information, each master can determine whether and when there is enough time to send a request packet (as described above under Protocol) before the bus becomes busy with another data block transfer and whether the corresponding data block transfer will interfere with pending bus transactions. Thus each master must read every request packet and update its bus-busy data structure to maintain information about when the bus is and will be free.

[0066] With two or more masters on the bus, masters will occasionally transmit independent request packets during the same bus cycle. Those multiple requests will collide as each such master drives the bus simultaneously with different information, resulting in scrambled request information and neither desired data block transfer. In a preferred form of the invention, each device on the bus seeking to write a logical 1 on a BusData or AddrValid line drives that line with a current sufficient to sustain a voltage greater than or equal to the high-logic value for the system. Devices do not drive lines that should have a logical 0; those lines are simply held at a voltage corresponding to a low-logic value. Each master tests the voltage on at least some, preferably all, bus data and the AddrValid lines so the master can detect a logical '1' where the expected level is '0' on a line that it does not drive during a given bus cycle but another master does drive.

[0067] Another way to detect collisions is to select one or more bus lines for collision signalling. Each master sending a request drives that line or lines and monitors the selected lines for more than the normal drive current (or a logical value of ">1"), indicating requests by more than one master. Persons skilled in the art will recognize that this can be implemented with a protocol involving BusData and AddrValid lines or could be implemented using an additional bus line.

[0068] In the preferred form of this invention, each master detects collisions by monitoring lines which it does not drive to see if another master is driving those lines. Referring to Fig. 4, the first byte of the request packet includes the number of each master attempting to use the bus (Master[0:3]). If two masters send packet requests starting at the same point in time, the master numbers will be logical "or"ed together by at least those masters, and thus one or both of the masters, by monitoring the data on the bus and comparing what it sent, can detect a collision. For instance if requests by masters number 2 (0010) and 5 (0101) collide, the bus will be driven with the value Master[0:3]=7 (0010 + 0101 = 0111). Master number 5 will detect that the signal Master[2] = 1 and master 2 will detect that Master[1] and Master[3] = 1, telling both masters that a collision has occurred. Another example is masters 2 and 11, for which the bus will be driven with the value Master[0:3]=11 (0010 + 1011 = 1011), and although master 11 can't readily detect this collision, master 2 can. When any collision is detected, each master detecting a collision drives the value of AddrValid 27 in byte 5 of the request packet 22 to 1, which is detected by all masters, including master 11 in the second example above, and forces a bus arbitration cycle, described below.

[0069] Another collision condition may arise where master A sends a request packet in cycle 0 and master B tries to send a request packet starting in cycle 2 of the first request packet, thereby overlapping the first request packet. This will occur from time to time because the bus operates at high speeds, thus the logic in a second-initiating master may not be fast enough to detect a request initiated by a first master in cycle 0 and to react fast enough by delaying its own request. Master B eventually notices that it wasn't supposed to try to send a request packet (and consequently almost surely destroyed the address that master A was trying to send), and, as in the example above of a simultaneous collision, drives a 1 on AddrValid during byte 5 of the first request packet 27 forcing an arbitration. The logic in the preferred implementation is fast enough that a master should detect a request packet by another master by cycle 3 of the first request packet, so no master is likely to attempt to send a potentially colliding request packet later than cycle 2.

[0070] Slave devices not need to detect a collision directly, but they must wait to do anything irrecoverable until the last byte (byte 5) is read to ensure that the packet is valid. A request packet with Master[0:3] equal to 0 (a retry signal) is ignored and does not cause a collision. The subsequent bytes of such a packet are ignored.

[0071] To begin arbitration after a collision, the masters wait a preselected number of cycles after the aborted request packet (4 cycles in a preferred implementation), then use the next free cycle to arbitrate for the bus (the next available even cycle in the preferred implementation). Each colliding master signals to all other colliding masters that it seeks to send a request packet, a priority is assigned to each of the colliding masters, then each master is allowed to make its request in the order of that priority.

[0072] Figure 6 illustrates one preferred way of implementing this arbitration. Each colliding master signals its intent to send a request packet by driving a single BusData line during a single bus cycle corresponding to its assigned master number (1-15 in the present example). During two-byte arbitration cycle 29, byte 0 is allocated to requests 1-7 from masters 1-7, respectively, (bit 0 is not used) and byte 1 is allocated to requests 8-15 from masters 8-15, respectively. At least one device and preferably each colliding master reads the values on the bus during the arbitration cycles to determine and store which masters desire to use the bus. Persons skilled in the art will recognize that a single byte can be allocated for arbitration requests if the system includes more bus lines than masters. More than 15 masters can be accommodated by using additional bus cycles.

[0073] A fixed priority scheme (preferably using the master numbers, selecting lowest numbers first) is then used to prioritize, then sequence the requests in a bus arbitration queue which is maintained by at least one device. These requests are queued by each master in the bus-busy data structure and no further requests are allowed until the bus

arbitration queue is cleared. Persons skilled in the art will recognize that other priority schemes can be used, including assigning priority according to the physical location of each master.

System Configuration/Reset

[0074] In the bus-based system of this invention, a mechanism is provided to give each device on the bus a unique device identifier (device ID) after power-up or under other conditions as desired or needed by the system. A master can then use this device ID to access a specific device, particularly to set or modify registers of the specified device, including the control and address registers. In the preferred embodiment, one master is assigned to carry out the entire system configuration process. The master provides a series of unique device ID numbers for each unique device connected to the bus system. In the preferred embodiment, each device connected to the bus contains a special device-type register which specifies the type of device, for instance CPU, 4 MBit memory, 64 MBit memory or disk controller. The configuration master should check each device, determine the device type and set appropriate control registers, including access-time registers. The configuration master should check each memory device and set all appropriate memory address registers.

[0075] One means to set up unique device ID numbers is to have each device to select a device ID in sequence and store the value in an internal device ID register. For example, a master can pass sequential device ID numbers through shift registers in each of a series of devices, or pass a token from device to device whereby the device with the token reads in device ID information from another line or lines. In a preferred embodiment, device ID numbers are assigned to devices according to their physical relationship, for instance, their order along the bus.

[0076] In a preferred embodiment of this invention, the device ID setting is accomplished using a pair of pins on each device, ResetIn and ResetOut. These pins handle normal logic signals and are used only during device ID configuration. On each rising edge of the clock, each device copies ResetIn (an input) into a four-stage reset shift register. The output of the reset shift register is connected to ResetOut, which in turn connects to ResetIn for the next sequentially connected device. Substantially all devices on the bus are thereby daisy-chained together. A first reset signal, for example, while ResetIn at a device is a logical 1, or when a selected bit of the reset shift register goes from zero to non-zero, causes the device to hard reset, for example by clearing all internal registers and resetting all state machines. A second reset signal, for example, the falling edge of ResetIn combined with changeable values on the external bus, causes that device to latch the contents of the external bus into the internal device ID register (Device[0:7]).

[0077] To reset all devices on a bus, a master sets the ResetIn line of the first device to a "1" for long enough to ensure that all devices on the bus have been reset (4 cycles times the number of devices -- note that the maximum number of devices on the preferred bus configuration is 256 (8 bits), so that 1024 cycles is always enough time to reset all devices.) Then ResetIn is dropped to "0" and the BusData lines are driven with the first followed by successive device ID numbers, changing after every 4 clock pulses. Successive devices set those device ID numbers into the corresponding device ID register as the falling edge of ResetIn propagates through the shift registers of the daisy-chained devices. Figure 14 shows ResetIn at a first device going low while a master drives a first device ID onto the bus data lines BusData[0:3]. The first device then latches in that first device ID. After four clock cycles, the master changes BusData[0:3] to the next device ID number and ResetOut at the first device goes low, which pulls ResetIn for the next daisy-chained device low, allowing the next device to latch in the next device ID number from BusData[0:3]. In the preferred embodiment, one master is assigned device ID 0 and it is the responsibility of that master to control the ResetIn line and to drive successive device ID numbers onto the bus at the appropriate times. In the preferred embodiment, each device waits two clock cycles after ResetIn goes low before latching in a device ID number from BusData[0:3].

[0078] Persons skilled in the art recognize that longer device ID numbers could be distributed to devices by having etch device read in multiple bytes from the bus and latch the values into the device ID register. Persons skilled in the art also recognize that there are alternative ways of getting device ID numbers to unique devices. For instance, a series of sequential numbers could be clocked along the ResetIn line and at a certain time each device could be instructed to latch the current reset shift register value into the device ID register.

[0079] The configuration master should choose and set an access time in each access-time register in each slave to a period sufficiently long to allow the slave to perform an actual, desired memory access. For example, for a normal DRAM access, this time must be longer than the row address strobe (RAS) access time. If this condition is not met, the slave may not deliver the correct data. The value stored in a slave access-time register is preferably one-half the number of bus cycles for which the slave device should wait before using the bus in response to a request. Thus an access time value of '1' would indicate that the slave should not access the bus until at least two cycles after the last byte of the request packet has been received. The value of AccessReg0 is preferably fixed at 8 (cycles) to facilitate access to control registers.

[0080] The bus architecture of this invention can include more than one master device. The reset or initialization sequence should also include a determination of whether there are multiple masters on the bus, and if so to assign unique master ID numbers to each. Persons skilled in the art will recognize that there are many ways of doing this. For

instance, the master could poll each device to determine what type of device it is, for example, by reading a special register then, for each master device, write the next available master ID number into a special register.

ECC

[0081] Error detection and correction ("ECC") methods well known in the art can be implemented in this system. ECC information typically is calculated for a block of data at the time that block of data is first written into memory. The data block usually has an integral binary size, e.g. 256 bits, and the ECC information uses significantly fewer bits. A potential problem arises in that each binary data block in prior art schemes typically is stored with the ECC bits appended, resulting in a block size that is not an integral binary power.

[0082] In a preferred embodiment of this invention, ECC information is stored separately from the corresponding data, which can then be stored in blocks having integral binary size. ECC information and corresponding data can be stored, for example, in separate DRAM devices. Data can be read without ECC using a single request packet, but to write or read error-corrected data requires two request packets, one for the data and a second for the corresponding ECC information. ECC information may not always be stored permanently and in some situations the ECC information may be available without sending a request packet or without a bus data block transfer.

[0083] In a preferred embodiment, a standard data block size can be selected for use with ECC, and the ECC method will determine the required number of bits of information in a corresponding ECC block. RAMs containing ECC information can be programmed to store an access time that is equal to: (1) the access time of the normal RAM (containing data) plus the time to access a standard data block (for corrected data) minus the time to send a request packet (6 bytes); or (2) the access time of a normal RAM minus the time to access a standard ECC block minus the time to send a request packet. To read a data block and the corresponding ECC block, the master simply issues a request for the data immediately followed by a request for the ECC block. The ECC RAM will wait for the selected access time then drive its data onto the bus right after (in case (1) above)) the data RAM has finished driving out the data block. Persons skilled in the art will recognize that the access time described in case (2) above can be used to drive ECC data before the data is driven onto the bus lines and will recognize that writing data can be done by analogy with the method described for a read. Persons skilled in the art will also recognize the adjustments that must be made in the bus-busy structure and the request packet arbitration methods of this invention in order to accommodate these paired ECC requests.

[0084] Since this system is quite flexible, the system designer can choose the size of the data blocks and the number of ECC bits using the memory devices of this invention. Note that the data stream on the bus can be interpreted in various ways. For instance the sequence can be 2^n data bytes followed by 2^m ECC bytes (or vice versa), or the sequence can be 2^k iterations of 8 data bytes plus 1 ECC byte. Other information, such as information used by a directory-based cache coherence scheme, can also be managed this way. *See*, for example, Anant Agarwal, et al., "Scalable Directory Schemes for Cache Consistency," 15th *International Symposium on Computer Architecture*, June 1988, pp. 280-289. Those skilled in the art will recognize alternative methods of implementing ECC schemes that are within the teachings of this invention.

Low Power 3-D Packaging

[0085] Another major advantage of this invention is that it drastically reduces the memory system power consumption. Nearly all the power consumed by a prior art DRAM is dissipated in performing row access. By using a single row access in a single RAM to supply all the bits for a block request (compared to a row-access in each of multiple RAMs in conventional memory systems) the power per bit can be made very small. Since the power dissipated by memory devices using this invention is significantly reduced, the devices potentially can be placed much closer together than with conventional designs.

[0086] The bus architecture of this invention makes possible an innovative 3-D packaging technology. By using a narrow, multiplexed (time-shared) bus, the pin count for an arbitrarily large memory device can be kept quite small - on the order of 20 pins. Moreover, this pin count can be kept constant from one generation of DRAM density to the next. The low power dissipation allows each package to be smaller, with narrower pin pitches (spacing between the IC pins). With current surface mount technology supporting pin pitches as low as 20 mils, all off-device connections can be implemented on a single edge of the memory device. Semiconductor die useful in this invention preferably have connections or pads along one edge of the die which can then be wired or otherwise connected to the package pins with wires having similar lengths. This geometry also allows for very short leads, preferably with an effective lead length of less than 4 mm. Furthermore, this invention uses only bused interconnections, i.e., each pad on each device is connected by the bus to the corresponding pad of each other device.

[0087] The use of a low pin count and an edge-connected bus permits a simple 3-D package, whereby the devices are stacked and the bus is connected along a single edge of the stack. The fact that all of the signals are bused is impor-

tant for the implementation of a simple 3-D structure. Without this, the complexity of the "backplane" would be too difficult to make cost effectively with current technology. The individual devices in a stack of the present invention can be packed quite tightly because of the low power dissipated by the entire memory system, permitting the devices to be stacked bumper-to-bumper or top to bottom. Conventional plastic-injection molded small outline (SO) packages can be used with a pitch of about 2.5 mm (100 mils), but the ultimate limit would be the device die thickness, which is about an order of magnitude smaller, 0.2-0.5 mm using current wafer technology.

Bus Electrical Description

[0088] By using devices with very low power dissipation and close physical packing, the bus can be made quite short, which in turn allows for short propagation times and high data rates. The bus of a preferred embodiment of the present invention consists of a set of resistor-terminated controlled impedance transmission lines which can operate up to a data rate of 500 MHz (2 ns cycles). The characteristics of the transmission lines are strongly affected by the loading caused by the DRAMs (or other slaves) mounted on the bus. These devices add lumped capacitance to the lines which both lowers the impedance of the lines and decreases the transmission speed. In the loaded environment, the bus impedance is likely to be on the order of 25 ohms and the propagation velocity about $c/4$ (c = the speed of light) or 7.5 cm/ns. To operate at a 2 ns data rate, the transit time on the bus should preferably be kept under 1 ns, to leave 1 ns for the setup and hold time of the input receivers (described below) plus clock skew. Thus the bus lines must be kept quite short, under about 8 cm for maximum performance. Lower performance systems may have much longer lines, e.g. a 4 ns bus may have 24 cm lines (3 ns transit time, 1 as setup and hold time).

[0089] In the preferred embodiment, the bus uses current source drivers. Each output must be able to sink 50 mA, which provides an output swing of about 500 mV or more. In the preferred embodiment of this invention, the bus is active low. The unasserted state (the high value) is preferably considered a logical zero, and the asserted value (low state) is therefore a logical 1. Those skilled in the art understand that the method of this invention can also be implemented using the opposite logical relation to voltage. The value of the unasserted state is set by the voltage on the termination resistors, and should be high enough to allow the outputs to act as current sources, while being as low as possible to reduce power dissipation. These constraints may yield a termination voltage about 2V above ground in the preferred implementation. Current source drivers cause the output voltage to be proportional to the sun of the sources driving the bus.

[0090] Referring to Fig.7, although there is no stable condition where two devices drive the bus at the same time, conditions can arise because of propagation delay on the wires where one device, A **41**, can start driving its part of the bus **44** while the bus is still being driven by another device, B **42** (already asserting a logical 1 on the bus). In a system using current drivers, when B **42** is driving the bus (before time **46**), the value at points **44** and **45** is logical 1. If B **42** switches off at time **46** just when A **41** switches on, the additional drive by device A **41** causes the voltage at the output **44** of A **41** to drop briefly below the normal value. The voltage returns to its normal value at time **47** when the effect of device B **42** turning off is felt. The voltage at point **45** goes to logical 0 when device B **42** turns off, then drops at time **47** when the effect of device A **41** tuning on is felt. Since the logical 1 driven by current from device A **41** is propagated irrespective of the previous value on the bus, the value on the bus is guaranteed to settle after one time of flight (t_f) delay, that is, the time it takes a signal to propagate from one end of the bus to the other. If a voltage drive was used (as in ECL wired-ORing), a logical 1 on the bus (from device B **42** being previously driven) would prevent the transition put out by device A **41** being felt at the most remote part of the system, e.g., device **43**, until the turnoff waveform from device B **42** reached device A **41** plus one time of flight delay, giving a worst case settling time of twice the time of flight delay.

Clocking

[0091] Clocking a high speed bus accurately without introducing error due to propagation delays can be implemented by having each device monitor two bus clock signals and then derive internally a device clock, the true system clock. The bus clock information can be sent on one or two lines to provide a mechanism for each bused device to generate an internal device clock with zero skew relative to all the other device clocks. Referring to Figure 8, in the preferred implementation, a bus clock generator **50** at one end of the bus propagates an early bus clock signal in one direction along the bus, for example on line **53** from left to right, to the far end of the bus. The same clock signal then is passed through the direct connection shown to a second line **54**, and returns as a late bus clock signal along the bus from the far end to the origin, propagating from right to left. A single bus clock line can be used if it is left unterminated at the far end of the bus, allowing the early bus clock signal to reflect back along the same line as a late bus clock signal.

[0092] Figure 8b illustrates how each device **51, 52** receives each of the two bus clock signals at a different time (because of propagation delay along the wires), with constant midpoint in time between the two bus clocks along the bus. At each device **51, 52**, the rising edge **55** of Clock1 **53** is followed by the rising edge **56** of Clock2 **54**. Similarly, the falling edge **57** of Clock1 **53** is followed by the falling edge **58** of Clock2 **54**. This waveform relationship is observed at

all other devices along the bus. Devices which are closer to the clock generator have a greater separation between Clock1 and Clock2 relative to devices farther from the generator because of the longer time required for each clock pulse to traverse the bus and return along line **54**, but the midpoint in time **59**, **60** between corresponding rising or falling edges is fixed because, for any given device, the length of each clock line between the far end of the bus and that device is equal. Each device must sample the two bus clocks and generate its own internal device clock at the midpoint of the two.

[0093] Clock distribution problems can be further reduced by using a bus clock and device clock rate equal to the bus cycle data rate divided by two, that is, the bus clock period is twice the bus cycle period. Thus a 500 MHz bus preferably uses a 250 MHz clock rate. This reduction in frequency provides two benefits. First it makes all signals on the bus have the same worst case data rates -- data on a 500 MHz bus can only change every 2 ns. Second, clocking at half the bus cycle data rate makes the labeling of the odd and even bus cycles trivial, for example, by defining even cycles to be those when the internal device clock is 0 and odd cycles when the internal device clock is 1.

Multiple Buses

[0094] The limitation on bus length described above restricts the total number of devices that can be placed on a single bus. Using 2.5 mm spacing between devices, a single 8 cm bus will hold about 32 devices. Persons skilled in the art will recognize certain applications of the present invention wherein the overall data rate on the bus is adequate but memory or processing requirements necessitate a much larger number of devices (many more than 32). Larger systems can easily be built using the teachings of this invention by using one or more memory subsystems, designated primary bus units, each of which consists of two or more devices, typically 32 or close to the maximum allowed by bus design requirements, connected to a transceiver device.

[0095] Referring to Figure 9, each primary bus unit can be mounted on a single circuit board **66**, sometimes called a memory stick. Each transceiver device **19** in turn connects to a transceiver bus **65**, similar or identical in electrical and other respects to the primary bus **18** described at length above. In a preferred implementation, all masters are situated on the transceiver bus so there are no transceiver delays between masters and all memory devices are on primary bus units so that all memory accesses experience an equivalent transceiver delay, but persons skilled in the art will recognize how to implement systems which have masters on more than one bus unit and memory devices on the transceiver bus as well as on primary bus units. In general, each teaching of this invention which refers to a memory device can be practiced using a transceiver device and one or more memory devices on an attached primay bus unit. Other devices, generically referred to as peripheral devices, including disk controllers, video controllers or I/O devices can also be attached to either the transceiver bus or a primary bus unit, as desired. Persons skilled in the art will recognize how to use a single primary bus unit or multiple primary bus units as needed with a transceiver bus in certain system designs.

[0096] The transceivers are quite simple in function. They detect request packets on the transceiver bus and transmit them to their primary bus unit. If the request packet calls for a write to a device on a transceiver's primary bus unit, that transceiver keeps track of the access time and block size and forwards all data from the transceiver bus to the primary bus unit during that time. The transceivers also watch their primary bus unit, forwarding any data that occurs there to the transceiver bus. The high speed of the buses means that the transceivers will need to be pipelined, and will require an additional one or two cycle delay for data to pass through the transceiver in either direction. Access times stored in masters on the transceiver bus must be increased to account for transceiver delay but access times stored in slaves on a primary bus unit should not be modified.

[0097] Persons skilled in the art will recognize that a more sophisticated transceiver can control transmissions to and from primary bus units. An additional control line, TrncvrRW can be bused to all devices on the transceiver bus, using that line in conjunction with the AddrValid line to indicate to all devices on the transceiver bus that the information on the data lines is: 1) a request packet, 2) valid data to a slave, 3) valid data from a slave, or 4) invalid data (or idle bus). Using this extra control line obviates the need for the transceivers to keep track of when data needs to be forwarded from its primary bus to the transceiver bus - all transceivers send all data from their primary bus to the transceiver bus whenever the control signal indicates condition 2) above. In a preferred implementation of this invention, if AddrValid and TrncvrRW are both low, there is no bus activity and the transceivers should remain in an idle state. A controller sending a request packet will drive AddrValid high, indicating to all devices on the transceiver bus that a request packet is being sent which each transceiver should forward to its primary bus unit. Each controller seeking to write to a slave should drive both AddrValid and TrncvrRW high, indicating valid data for a slave is present on the data lines. Each transceiver device will then transmit all data from the transceiver bus lines to each primary bus unit. Any controller expecting to receive information from a slave should also drive the TrncvrRW line high, but not drive AddrValid, thereby indicating to each transceiver to transmit any data coming from any slave on its primary local bus to the transceiver bus. A still more sophisticated transceiver would recognize signals addressed to or coming from its primary bus unit and transmit signals only at requested times.

[0098] An example of the physical mounting of the transceivers is shown in Figure 9. One important feature of this

physical arrangement is to integrate the bus of each transceiver **19** with the original bus of DRAMs or other devices **15, 16, 17** on the primary bus unit **66**. The transceivers **19** have pins on two sides, and are preferably mounted flat on the primary bus unit with a first set of pins connected to primary bus **18**. A second set of transceiver pins **20**, preferably orthogonal to the first set of pins, are oriented to allow the transceiver **19** to be attached to the transceiver bus **65** in much the same way as the DRAMs were attached to the primary bus unit. The transceiver bus can be generally planar and in a different plane, preferably orthogonal to the plane of each primary bus unit. The transceiver bus can also be generally circular with primary bus units mounted perpendicular and tangential to the transceiver bus.

[0099] Using this two level scheme allows one to easily build a system that contains over 500 slaves (16 buses of 32 DRAMs each). Persons skilled in the art can modify the device ID scheme described above to accommodate more than 256 devices, for example by using a longer device ID or by using additional registers to hold some of the device ID. This scheme can be extended in yet a third dimension to make a second-order transceiver bus, connecting multiple transceiver buses by aligning transceiver bus units parallel to and on top of each other and busing corresponding signal lines through a suitable transceiver. Using such a second-order transceiver bus, one could connect many thousands of slave devices into what is effectively a single bus.

Device Interface

[0100] The device interface to the high-speed bus can be divided into three main parts. The first part is the electrical interface. This part includes the input receivers, bus drivers and clock generation circuitry. The second part contains the address comparison circuitry and timing registers. This part takes the input request packet and determines if the request is for this device, and if it is, starts the internal access and delivers the data to the pins at the correct time. The final part, specifically for memory devices such as DRAMs, is the DRAM column access path. This part needs to provide bandwidth into and out of the DRAM sense amps greater than the bandwidth provided by conventional DRAMs. The implementation of the electrical interface and DRAM column access path are described in more detail in the following sections. Persons skilled in the art recognize how to modify prior-art address comparison circuitry and prior-art register circuitry in order to practice the present invention.

Electrical Interface - Input/Output Circuitry

[0101] A block diagram of the preferred input/output circuit for address/data/control lines is shown in Figure 10. This circuitry is particularly well-suited for use in DRAM devices but it can be used or modified by one skilled in the art for use in other devices connected to the bus of this invention. It consists of a set of input receivers **71, 72** and output driver **76** connected to input/output line **69** and pad **75** and circuitry to use the internal clock **73** and internal clock complement **74** to drive the input interface. The clocked input receivers take advantage of the synchronous nature of the bus. To further reduce the performance requirements for device input receivers, each device pin, and thus each bus line, is connected to two clocked receivers, one to sample the even cycle inputs, the other to sample the odd cycle inputs. By thus de-multiplexing the input **70** at the pin, each clocked amplifier is given a full 2 ns cycle to amplify the bus low-voltage-swing signal into a full value CMOS logic signal. Persons skilled in the art will recognize that additional clocked input receivers can be used within the teachings of this invention. For example, four input receivers could be connected to each device pin and clocked by a modified internal device clock to transfer sequential bits from the bus to internal device circuits, allowing still higher external bus speeds or still longer settling times to amplify the bus low-voltage-swing signal into a full value CMOS logic signal.

[0102] The output drivers are quite simple, and consist of a single NMOS pulldown transistor **76**. This transistor is sized so that under worst case conditions it can still sink the 50 mA required by the bus. For 0.8 micron CMOS technology, the transistor will need to be about 200 microns long. Overall bus performance can be improved by using feedback techniques to control output transistor current so that the current through the device is roughly 50 mA under all operating conditions, although this is not absolutely necessary for proper bus operation. An example of one of many methods known to persons skilled in the art for using feedback techniques to control current is described in Hans Schumacher, et al.., "CMOS Subnanosecond True-ECL Output Buffer," *J. Solid State Circuits*, Vol. 25 (1), pp. 150-154 (Feb. 1990). Controlling this current improves performance and reduces power dissipation. This output driver which can be operated at 500 MHz, can in turn be controlled by a suitable multiplexer with two or more (preferably four) inputs connected to other internal chip circuitry, all of which can be designed according to well known prior art.

[0103] The input receivers of every slave must be able to operate during every cycle to determine whether the signal on the bus is a valid request packet. This requirement leads to a number of constraints on the input circuitry. In addition to requiring small acquisition and resolution delays, the circuits must take little or no DC power, little AC power and inject very little current back into the input or reference lines. The standard clocked DRAM sense amp shown in Figure 11 satisfies all these requirements except the need for low input currents. When this sense amp goes from sense to sample, the capacitance of the internal nodes **83** and **84** in Figure11 is discharged through the reference line **68** and

input **69**, respectively. This particular current is small, but the sum of such currents from all the inputs into the reference lines summed over all devices can be reasonably large.

[0104] The fact that the sign of the current depends upon on the previous received data makes matters worse. One way to solve this problem is to divide the sample period into two phases. During the first phase, the inputs are shorted to a buffered version of the reference level (which may have an offset). During the second phase, the inputs are connected to the true inputs. This scheme does not remove the input current completely, since the input must still charge nodes **83** and **84** from the reference value to the current input value, but it does reduce the total charge required by about a factor of 10 (requiring only a 0.25V change rather than a 2.5V change). Persons skilled in the art will recognize that many other methods can be used to provide a clocked amplifier that will operate on very low input currents.

[0105] One important part of the input/output circuitry generates an internal device clock based on early and late bus clocks. Controlling clock skew (the difference in clock timing between devices) is important in a system running with 2 ns cycles, thus the internal device clock is generated so the input sampler and the output driver operate as close in time as possible to midway between the two bus clocks.

[0106] A block diagram of the internal device clock generating circuit is shown in Figure 12 and the corresponding timing diagram in Figure 13. The basic idea behind this circuit is relatively simple. A DC amplifier **102** is used to convert the small-swing bus clock into a full-swing CMOS signal. This signal is then fed into a variable delay line **103**. The output of delay line **103** feeds three additional delay lines: **104** having a fixed delay; **105** having the same fixed delay plus a second variable delay; and **106** having the same fixed delay plus one half of the second variable delay. The outputs **107**, **108** of the delay lines **104** and **105** drive clocked input receivers **101** and **111** connected to early and late bus clock inputs **100** and **110**, respectively. These input receivers **101** and **111** have the same design as the receivers described above and shown in Fig. 11. Variable delay lines **103** and **105** are adjusted via feedback lines **116**, **115** so that input receivers **101** and **111** sample the bus clocks just as they transition. Delay lines **103** and **105** are adjusted so that the falling edge **120** of output **107** precedes the falling edge **121** of the early bus clock, Clock1 **53**, by an amount of time **128** equal to the delay in input sampler **101**. Delay line **108** is adjusted in the same way so that falling edge **122** precedes the falling edge **123** of late bus clock, Clock2 **54**, by the delay **128** in input sampler **111**.

[0107] Since the outputs **107** and **108** are synchronized with the two bus clocks and the output **73** of the last delay line **106** is midway between outputs **107** and **108**, that is, output **73** follows output **107** by the same amount of time **129** that output **73** precedes output **108**, output **73** provides an internal device clock midway between the bus clocks. The falling edge **124** of internal device clock **73** precedes the time of actual input sampling **125** by one sampler delay. Note that this circuit organization automatically balances the delay in substantially all device input receivers **71** and **72** (Fig. 10), since outputs **107** and **108** are adjusted so the bus clocks are sampled by input receivers **101** and **111** just as the bus clocks transition.

[0108] In the preferred embodiment, two sets of these delay lines are used, one to generate the true value of the internal device clock **73**, and the other to generate the complement **74** without adding any inverter delay. The dual circuit allows generation of truly complementary clocks, with extremely small skew. The complement internal device clock is used to clock the 'even' input receivers to sample at time **127**, while the true internal device clock is used to clock the 'odd' input receivers to sample at time **125**. The true and complement internal device clocks are also used to select which data is driven to the output drivers. The gate delay between the internal device clock and output circuits driving the bus is slightly greater than the corresponding delay for the input circuits, which means that the new data always will be driven on the bus slightly after the old data has been sampled.

DRAM Column Access Modification

[0109] A block diagram of a conventional 4 MBit DRAM 130 is shown in Figure 15. The DRAM memory array is divided into a number of subarrays **150-157**, for example, 8. Each subarray is divided into arrays **148**, **149** of memory cells. Row address selection is performed by decoders **146**. A column decoder **147A**, **147B**, including column sense amps on either side of the decoder, runs through the core of each subarray. These column sense amps can be set to precharge or latch the most-recently stored value, as described in detail above. Internal I/O lines connect each set of sense-amps, as gated by corresponding column decoders, to input and output circuitry connected ultimately to the device pins. These internal I/O lines are used to drive the data from the selected bit lines to the data pins (some of pins **131-145**), or to take the data from the pins and write the selected bit lines. Such a column access path organized by prior art constraints does not have sufficient bandwidth to interface with a high speed bus. The method of this invention does not require changing the overall method used for column access, but does change implementation details. Many of these details have been implemented selectively in certain fast memory devices, but never in conjunction with the bus architecture of this invention.

[0110] Running the internal I/O lines in the conventional way at high bus cycle rates is not possible. In the preferred method, several (preferably 4) bytes are read or written during each cycle and the column access path is modified to run at a lower rate (the inverse of the number of bytes accessed per cycle, preferably 1/4 of the bus cycle rate). Three

different techniques are used to provide the additional internal I/O lines required and to supply data to memory cells at this rate. First, the number of I/O bit lines in each subarray running through the column decoder **147** is increased, for example, to 16, eight for each of the two columns of column sense amps and the column decoder selects one set of columns from the "top" half **148** of subarray **150** and one set of columns from the "bottom" half **149** during each cycle, where the column decoder selects one column sense amp per I/O bit line. Second, each column I/O line is divided into two halves, carrying data independently over separate internal I/O lines from the left half **147A** and right half **147B** of each subarray (dividing each subarray into quadrants) and the column decoder selects sense amps from each right and left half of the subarray, doubling the number of bits available at each cycle. Thus each column decode selection turns on *n* column sense amps, where *n* equals four (top left and right, bottom left and right quadrants) times the number of I/O lines in the bus to each subarray quadrant (8 lines each x 4=32 lines in the preferred implementation). Finally, during each RAS cycle, two different subarrays, e.g. 157 and 153, are accessed. This doubles again the available number of I/O lines containing data. Taken together, these changes increase the internal I/O bandwidth by at least a factor of 8. Four internal buses are used to route these internal I/O lines. Increasing the number of I/O lines and then splitting them in the middle greatly reduces the capacitance of each internal I/O line which in turn reduces the column access time, increasing the column access bandwidth even further.

[0111] The multiple, gated input receivers described above allow high speed input from the device pins onto the internal I/O lines and ultimately into memory. The multiplexed output driver described above is used to keep up with the data flow available using these techniques. Control means are provided to select whether information at the device pins should be treated as an address, and therefore to be decoded, or input or output data to be driven onto or read from the internal I/O lines.

[0112] Each subarray can access 32 bits per cycle, 16 bit from the left subarray and 16 from the right subarray. With 8 I/O lines per sense-amplifier column and accessing two subarrays at a time, the DRAM can provide 64 bits per cycle. This extra I/O bandwidth is not needed for reads (and is probably not used), but may be needed for writes. Availability of write bandwidth is a more difficult problem than read bandwidth because over-writing a value in a sense-amplifier may be a slow operation, depending on how the sense amplifier is connected to the bit line. The extra set of internal I/O lines provides some bandwidth margin for write operations.

[0113] Persons skilled in the art will recognize that many variations of the teachings of this invention can be practiced that still fall within the claims of this invention which follow.

Claims

1. A memory subsystem comprising

   two memory devices connected in parallel to a bus,
   said bus including a plurality of bus lines for carrying substantially all address, data and control information needed by said memory devices,
   said control information including device-select information,
   said bus containing substantially fewer bus lines than the number of bits in a single address, and
   said bus carrying device-select information without the need for separate device-select lines connected directly to individual memory devices.

2. The memory subsystem with the features of embodiment 1 wherein said bus contains at least 8 bus lines adapted to carry at least 16 address bits and at least 8 data bits.

3. The memory subsystem with the features of embodiment 1 wherein said bus also includes parallel lines for clock and power.

4. A system comprising

   a memory subsystem with the features of embodiment 1 wherein each bus of said memory subsystem is connected to its own transceiver device,
   a transceiver bus connecting said transceiver devices, and
   a means for transferring information between each of said buses of said memory subsystems and said transceiver bus, whereby memory subsystems may be integrated into a larger system having more memory than an individual memory subsystem.

5. The system with the features of embodiment 4 having a plurality of memory subsystems.

6. The system with the features of embodiment 4 further comprising a master device connected to said transceiver bus.

7. The system with the features of embodiment 6 wherein said master device is selected from the group consisting of a central processing unit, a floating point unit and a direct memory access unit.

8. The system with the features of embodiment 4 further comprising a peripheral device connected to the transceiver bus, said peripheral device adapted for connection to other devices not on the bus.

9. The system with the features of embodiment 8 said peripheral device is selected from the group consisting of an I/O interface port, a video controller and a disk controller.

10. The system with the features of embodiment 5 wherein said transceiver bus is in a different plane than the plane of the bus of each of said memory subsystems.

11. The system with the features of embodiment 5 wherein the bus of each memory subsystem lies substantially in a subsystem bus plane and said transceiver bus lies substantially in a plane orthogonal to said subsystem bus plane.

12. The system with the features of embodiment 4 having at least two transceiver buses, each transceiver bus having a plurality of memory subsystem buses connected through a first transceiver to said transceiver bus, each of said transceiver buses being further connected to a second transceiver adapted to interface to a second-order transceiver bus, whereby each transceiver bus is connected through said second transceiver to form a second-order transceiver bus unit.

13. A semiconductor subsystem bus for interconnecting semiconductor devices comprising

a plurality of semiconductor devices connected in parallel to a bus, at least one of said semiconductor devices being a memory device or a transceiver device which in turn is connected to a memory subsystem,
said bus including a plurality of bus lines for carrying substantially all address, data and control information needed by said semiconductor devices,
said control information including semiconductor device-select information,
said bus containing substantially fewer bus lines than the number of bits in a single address, and
said bus carrying device-select information without the need for separate device-select lines connected directly to individual semiconductor devices, and
at least one modifiable register in each of the semi-conductor devices on said bus, said modifiable registers being accessible from said bus, whereby the subsystem can be configured using signals transmitted on said bus.

14. The semiconductor subsystem bus with the features of embodiment 13 wherein one type of modifiable register is an access-time register designed to store a time delay after which a device may take some specified action on said bus.

15. The semiconductor subsystem bus with the features of embodiment 13 further comprising a semiconductor device having at least two access-time registers and
one of said access-time registers is permanently programmed to contain a fixed value and at least one of said access-time registers can be modified by information carried on said bus.

16. The semiconductor subsystem bus with the features of embodiment 13 further 13 further comprising a memory device having at least one discrete memory section and also having a modifiable address register adapted to store memory address information which corresponds to each said discrete memory section.

17. The semiconductor subsystem bus with the features of embodiment 16 wherein said memory address information comprises a pointer to said discrete memory section.

18. The semiconductor subsystem bus with the features of embodiment 16 wherein said discrete memory section has a top and a bottom and said memory address information comprises pointers to said top and said bottom.

19. The semiconductor subsystem bus with the features of embodiment 16 wherein said memory address information

comprises

a pointer to said discrete memory section and
a range value indicating the size of said discrete memory section.

20. The semiconductor subsystem bus with the features of embodiment 16 wherein said address registers of each of said discrete memory sections of each of said memory devices connected to said bus are set to contain memory address information that is different for each discrete memory section and such that the highest memory address in each discrete memory section is one less than the lowest memory address in another discrete memory section, whereby memory may be organized into one or a small number of contiguous memory blocks.

21. The semiconductor subsystem bus with the features of embodiment 16 further comprising a means for testing each of said discrete memory sections of each of said memory devices for proper function, and
for each non-functional discrete memory section, a means for setting at least one address register which corresponds to said discrete memory section to indicate that said discrete memory section is non-functional,
for each functional discrete memory section, a means for setting at least one address register which corresponds to said discrete memory section to contain such corresponding address information.

22. The semiconductor subsystem bus with the features of embodiment 21 wherein said address registers corresponding to said discrete memory sections are set to provide one contiguous memory block within the subsystem.

23. The semiconductor subsystem bus with the features of embodiment 13 wherein one of said modifiable registers is a device identification register which can be modified to contain a value unique to that semiconductor device.

24. The semiconductor subsystem bus with the features of embodiment 23 wherein said device identification register is set to contain a unique value which is a function of the physical position of that semiconductor device either along said bus or in relationship to other semiconductor devices or said bus.

25. A bus subsystem comprising

two semiconductor devices connected in parallel to a bus, wherein one of said semiconductor devices is a master device,
said master device including a means for initiating bus transactions,
said bus including a plurality of bus lines for carrying substantially all address, data and control information needed by said devices,
said control information including device-select information,
said bus containing substantially fewer lines than the number of bits in a single address, and
said bus carrying device-select information without the need for separate device-select lines connected directly to individual devices on said bus, whereby said master device initiates bus transactions which transfer information between said semiconductor devices on said bus.

26. The bus subsystem with the features of embodiment 25 wherein one of said semiconductor devices is a memory device connected to said bus, said memory device having at least one discrete memory section and also having a modifiable address register adapted to store memory address information which corresponds to each said discrete memory section.

27. The bus subsystem with the features of embodiment 26 wherein one of said semiconductor devices comprises a transceiver device connected in parallel to said bus and connected in parallel to a memory device on a bus other than said bus.

28. The bus subsystem with the features of embodiment 26 further including a means for said master device to request said memory device to prepare for a bus transaction by sending a request packet along said bus, said memory device and said master device each having a device-internal means to prepare to begin said bus transaction during a device-internal phase and further having a bus access means to effect said bus transaction during a bus access phase, said request packet including

a sequence of bytes containing address and control information,
said control information including information about the requested bus transaction and about the access time,

which corresponds to a number of bus cycles, which needs to intervene before beginning said bus-access phase, and
said address information pointing to at least one memory location within one of said discrete memory sections of said memory device.

29. The bus subsystem with the features of embodiment 28 wherein said memory device includes a means to read said control information and initiate said device-internal means at a time so as to complete said device-internal phase within said access time and begin said bus access phase after said number of bus cycles.

30. The bus subsystem with the features of embodiment 28 wherein said control information comprises an op code.

31. The bus subsystem with the features of embodiment 30 wherein said memory device includes sense amplifiers adapted to hold a bit of information or to precharge after a selected time and a means to transfer a data block during a data block transfer either reading data from said memory device or writing data into said memory device, and wherein said op code instructs said memory device to activate a response means, said response means including a means to

initiate a data block transfer,
select the size of said data block,
select the time to initiate said data block transfer,
access a control register, including reading from or writing to said control register,
precharge said sense amplifiers after each of said data block transfers is complete,
hold a bit of information in each of said sense amplifiers after each of said data block transfers is complete, or
select normal or page-mode access.

32. The bus subsystem with the features of embodiment 31 wherein said data block transfer comprises a read from or a write to memory within a single memory device.

33. The bus subsystem with the features of embodiment 28 further comprising a means for said master device to send control information to a specific one of said semiconductor devices on said bus by including in said request packet a device identification number unique to said semiconductor device.

34. The bus subsystem with the features of embodiment 28 further comprising a means for said master device to send control information to a selected one of said discrete memory portions by including in said request packet a specific memory address.

35. The bus subsystem with the features of embodiment 28 further comprising a means for said master device to send control information to substantially all semiconductor devices on said bus by including in said request packet a special device identification number which is recognized by said semiconductor devices.

36. The bus subsystem with the features of embodiment 28 wherein said control information specifies directly or indirectly the number of bus cycles for said master device and said memory device to wait before beginning said bus access phase.

37. The bus subsystem with the features of embodiment 36 wherein, for a data block transfer, said master device and said memory device use the same access time and same data block size regardless of whether said data block transfer is a read or write operation.

38. The bus subsystem with the features of embodiment 28 wherein said control information further includes a block-size value that encodes and specifies the size of the block of data to be transferred.

39. The bus subsystem with the features of embodiment 38 wherein said block-size value is encoded as a linear value for relatively small block sizes values and is encoded as a logarithmic value for relatively larger block sizes.

40. The bus subsystem with the features of embodiment 38 wherein said block-size value is encoded using four bits, and where the encoded value is

Encoded Value	Block Size (Bytes)
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	16
10	32
11	64
12	128
13	256
14	512
15	1024

41. The bus subsystem with the features of embodiment 26 wherein said memory device is a DRAM device containing

a plurality of sense amplifiers,
a means to hold said sense amplifiers in an unmodified state after a read or write operation, leaving the device in page mode,
a means to precharge said sense amplifiers and
a means for selecting whether to precharge said sense amplifiers or to hold said sense amplifiers in an unmodified state.

42. The bus subsystem with the features of embodiment 28 wherein said request packet comprises an even number of bytes.

43. The bus subsystem with the features of embodiment 28 further including a means for generating and controlling a plurality of bus cycles, during which said bus carries said address, data and control information, and wherein alternate said bus cycles are designated odd cycles and even cycles, respectively, and wherein said request packet begins only on an even cycle.

44. The bus subsystem with the features of embodiment 28 further including a means for generating ECC information corresponding to a block of data and a means for using said ECC information to correct errors in storing or reading said block of data, wherein said ECC information may be stored separately from said block of data.

45. The bus subsystem with the features of embodiment 44 further comprising at least two of said memory devices wherein said ECC information and said corresponding block of data are stored in a first and a second said memory device, respectively, and said master device includes a means to write or read said block of data with error correction by sending separate ones of said request packets for said ECC information and for said corresponding block of data.

46. A bus subsystem comprising

a memory device and a master device connected in parallel on a bus,

a means for said master device to send a request packet and initiate a bus transaction and
a means for said master device to keep track of current and pending bus transactions,

said bus including a plurality of bus lines for carrying substantially all address, data and control information needed by said memory devices,
said bus containing substantially fewer lines than the number of bits in a single address, and
said bus carrying device-select information without the need for separate device-select lines connected directly to individual devices on said bus, whereby said master device initiates bus transactions which transfer information between devices on said bus and collisions on said bus are avoided because said master device avoids initiating bus transactions which would conflict with current or pending bus transactions.

47. The bus subsystem with the features of embodiment 46 having at least two of said master devices and including

a collision detecting means whereby a first said master device sending a first said request packet can detect a second said master device sending one of said colliding request packets, where one of said said colliding request packet may be sent simultaneous with the initial sending of or overlapping the sending of said first request packet, and

an arbitration means whereby said first and said second master devices select a priority order in which each of said master devices will be allowed to access said bus sequentially.

48. The bus subsystem with the features of embodiment 47 wherein each of said master devices has a master ID number and each of said request packets includes a master ID position which is a predetermined number of bits in a predetermined position in said request packet, and wherein said collision detection means comprises

a means included in each master device for sending a request packet including said master ID number of said master device in said master ID position of said request packet and
a means to detect a collision and invoke said arbitration means if any master device detects any other master ID number in said master ID position.

49. The bus subsystem with the features of embodiment 47 wherein each of said master devices includes

a means for sending a request packet,
a means for driving a selected bus line or lines during at least one selected bus cycle while said request packet is being sent,
a means for monitoring said selected bus line or lines to see if a said master device is sending a colliding request packet and
a means for informing all other master devices that a collision has occurred and for invoking said arbitration means.

50. The bus subsystem with the features of embodiment 47 wherein each of said master devices includes

a means, when sending a request packet, to drive a selected bus line or lines with a certain current during at least one selected bus cycle,
a means for monitoring said selected bus line or lines for a greater than normal current to see if another master device is driving that line or lines,
a means for detecting said greater than normal current, and
a means for informing all said master devices that a collision has occurred and for invoking said arbitration means.

51. The bus subsystem with the features of embodiment 47 wherein said arbitration means comprises

a means for initiating an arbitration cycle,
a means for allocating a single bus line to each master device during at least one selected bus cycle relative to the start of said arbitration cycle,
a means for allocating each master device to a single bus line during one of said selected bus cycles if there

are more master devices than available bus lines,

a means for each of said master devices which sent a colliding request packet to drive said bus line allocated to said master device during said selected bus cycle, and

a means in at least one of said master devices for storing information about which master devices sent a colliding request packet,

whereby said master devices can monitor selected bus lines during said arbitration cycle and identify each said master device which sent a colliding request packet.

52. The bus subsystem with the features of embodiment 47 wherein said arbitration means comprises

a means included in a first one of said master devices which sent colliding request packets for identifying each of said master devices which sent colliding request packets,

a means for assigning a priority to each said master device which sent a colliding request packet, and

a means for allowing each said master device which sent a colliding request packet to access the bus sequentially according to that priority.

53. The bus subsystem with the features of embodiment 52 wherein said priority is based on the physical location of each of said master devices.

54. The bus subsystem with the features of embodiment 52 wherein said priority is based on said master ID number of said master devices.

55. The bus subsystem with the features of embodiment 52 wherein each of said master devices includes a means, when sending a colliding request packet, for deciding which master device can send the next request packet in what order or at what time, whereby no master device may send a new request packet until responses to each pending request packet have been completed or scheduled.

56. A bus subsystem comprising

a plurality of semiconductor devices connected in parallel to a bus,

said bus including a plurality of bus lines for carrying substantially all address, data and control information needed by said semiconductor devices,

said control information including device-select information,

said bus containing substantially fewer lines than the number of bits in a single address,

said bus carrying said device-select information without the need for separate device-select lines connected directly to individual semiconductor devices,

said semiconductor devices including a reset means having an input and an output, the output of the reset means of one semiconductor device being connected to the input of the reset means of the next semiconductor device in series.

57. The bus subsystem with the features of embodiment 56 further including system reset means comprising

a means for generating a first and a second reset signal,

a means for passing said first reset signal to a first of said semiconductor devices and then to subsequent ones of said semiconductor devices in series and

a means for passing a second reset signal to said first semiconductor device and then to said subsequent semiconductor devices in series,

said bus subsystem including one of said semiconductor devices containing

a device identification register adapted to contain a number unique to said semiconductor device within said bus subsystem,

a device identification register setting means, and

a device reset means for resetting said semiconductor device to some desired, known reset state in response to said first reset signal and for setting said device identification register in response to said second reset signal,

whereby said bus subsystem can be reset to a known reset state with a unique device identification value in said device identification register of each of said semiconductor devices.

58. The bus subsystem with the features of embodiment 57 wherein said desired, known reset state is where all registers in the semiconductor device are cleared and the state machines are reset.

59. The bus subsystem with the features of embodiment 57 wherein said device identification register setting means comprises

a means for detecting said second reset signal,
a means for reading a device identification number from said bus lines at a specific time relative to said second reset signal and
a means for storing said device identification number in said device identification register of said semiconductor device.

60. The bus subsystem with the features of embodiment 57 wherein said second reset signal comprises multiple pulse sequences and wherein said device identification setting means includes

a means for interpreting said pulse sequences as a device identification number and
a means for storing said device identification number in said device identification register of said semiconductor device.

61. The bus subsystem with the features of embodiment 57 wherein said device reset means comprises an n-stage shift register capable of storing n-bit values, wherein said device reset means interprets a specific value in said shift register as said first reset signal and interprets a specific value in said shift register as said second reset signal.

62. The bus subsystem with the features of embodiment 57 wherein one of said semiconductor devices is a master device, said master device including a means for generating said first and said second reset signals.

63. The bus subsystem with the features of embodiment 57 wherein one of said semiconductor devices is a master device, said master device including

a master ID register,
a means for assigning a master ID number to said master device and
a means for storing said master ID number in said master ID register.

64. The bus subsystem with the features of embodiment 63 further comprising a second one of said master devices, and a means for a first one of said master devices to assign a master ID number to substantially all other said master devices, whereby said first master device assigns one of said master ID numbers to each of said master devices on said bus subsystem and each said master device stores said assigned master ID number in said master ID register.

65. The bus subsystem with the features of embodiment 57 wherein one of said semiconductor devices includes a device-type register adapted to contain an identifier characteristic of that type of semiconductor device, and one or more modifiable registers, at least one of which is an access-time register adapted for storing access times.

66. The bus subsystem with the features of embodiment 65 wherein one of said semiconductor devices is a master device having

a means for selecting a semiconductor device,
a means for reading said device-type register of said selected semiconductor device,
a means for determining the device type of said selected semiconductor device,
a means for determining access-time values appropriate for said selected semiconductor device and for storing said access-time values in said access-time registers of said selected semiconductor device, and
a means for selecting and storing other values appropriate for said selected semiconductor device in corresponding registers of said selected semiconductor device,
whereby said master device can select a semiconductor device, determine what type it is, and set said access-time and other registers to contain appropriate values.

67. The bus subsystem with the features of embodiment 66 further comprising a memory device having at least one discrete memory section and at least one modifiable address register adapted to store memory address informa-

tion which corresponds to each of said discrete memory sections, and
said master device further comprising a means for selecting and testing each of said discrete memory sections and a means for storing address information in said address registers corresponding to each of said discrete memory sections, whereby said master device can test all said discrete memory sections and assign unique address values thereto.

68. A bus subsystem comprising

two semiconductor devices connected in parallel to a bus, one of said semiconductor devices being a master device,
said bus including a plurality of bus data lines for carrying substantially all address, data and control information needed by said semiconductor devices,
said control information including device-select information,
said bus containing substantially fewer of said bus data lines than the number of bits in a single address, and
said bus carrying device-select information without the need for separate device-select lines connected directly to individual semiconductor devices,
wherein all of said bus data lines are terminated transmission lines and all of said address, data and control information is carried on said bus data lines as a sequential series of bits in the form of low-voltage-swing signals.

69. The bus subsystem with the features of embodiment 68 further comprising a semiconductor device including a current-mode driver connected to drive one of said bus data lines.

70. The bus subsystem with the features of embodiment 69 further comprising a semiconductor device having a means to measure the voltage of said low-voltage-swing signals on a selected one of said bus data lines, whereby said semiconductor device can determine whether zero, one, or more than one of said current-mode drivers are driving said selected bus data line.

71. The bus subsystem with the features of embodiment 70 further comprising a semiconductor device having

a plurality of input receivers connected to one of said bus data lines, and
a selection means for selecting said input receivers one by one to sense and store, one at a time, the bits of said sequential series of bits.

72. The bus subsystem with the features of embodiment 70 further comprising a semiconductor device having two input receivers connected to one of said bus data lines.

73. A bus subsystem comprising

two semiconductor devices connected in parallel to a bus having a first and a second end, said bus including a bus clock line, said bus clock line having first and second ends corresponding to said first and second ends of said bus, respectively,
a clock generator connected to said first end of said bus clock line to generate early bus clock signals with a normal rise time, and
signal return means at said second end of said bus clock line to return said early bus clock signals to said first end of said bus as corresponding late bus clock signals,
whereby each of said early bus clock signals will propagate from said clock generator along said clock line starting from said first end to said second end of said bus and then return at a later time to said first end of said bus as a corresponding late bus clock signal, whereby each semiconductor device on said bus can detect said early bus clock signals and said corresponding late bus clock signals.

74. The bus subsystem with the features of embodiment 73 further comprising a first and a second said bus clock line having first and second ends at said first and said second ends of said bus, respectively, wherein said signal return means directly connects said second ends of said first and said second bus clock lines whereby each of said early bus clock signals will propagate from said clock generator at said first end of said bus along said first bus clock line to said second end of said bus and then return on said second bus clock line to said first end of said bus as one of said corresponding late bus clock signals.

75. The bus subsystem with the features of embodiment 73 wherein said signal return means comprises said first bus clock line without a line terminator at said second end thereof whereby each of said early bus clock signals reaching said second end of said first bus clock line will be reflected back along said first bus clock line as said corresponding late bus clock signals.

76. The bus subsystem with the features of embodiment 73 further comprising

a means for operating said bus in bus cycles timed to have a certain bus cycle frequency and a corresponding bus cycle period and
a means for operating said clock generator with a period of twice the bus cycle period.

77. The bus subsystem with the features of embodiment 76 wherein said bus cycle frequency is greater than approximately 50 MHz and less than or equal to approximately 500 MHz.

78. The bus subsystem with the features of embodiment 73 further including a semiconductor device having an internal device clock generating means to derive the midpoint time between said early and corresponding late bus clock signals and to generate an internal device clock synchronized to said midpoint time.

79. The bus subsystem with the features of embodiment 73 further including a semiconductor device having a low-skew clock generator circuit comprising

a first delay line having an input, an output and a basic delay and means for synchronizing the output of said first delay line with said early bus clock signal,
a second delay line having said basic delay plus a variable delay, said second delay line having an output and
a means for synchronizing the output of said second delay line with said late bus clock signal, and
a third delay line having a third delay and a means to set said third delay midway between the delays of said first and second delay lines, said third delay line having an output which provides an internal device clock signal synchronized to a time halfway between said early and said late bus clock signals.

80. The bus subsystem with the features of embodiment 73 wherein said early and said late bus clock signals are low-voltage-swing signals that transition cyclically between low and high logical values, and further including a semiconductor device having a low-skew clock generator circuit comprising

a DC amplifier to convert said early and said late bus clock signals into full-swing logic signals,
a first variable delay line having a first variable delay and an input and an output, the input of said first variable delay line being connected to said DC amplifier
a first, a second and a third additional delay line, each having an input and an output, the input of each of said additional delay lines being connected to the output of said first delay line,
said first additional delay line having a fixed delay,
said second additional delay line having said fixed delay plus a second variable delay, and
said third additional delay line having said fixed delay plus one half of said second variable delay,
a first clocked input receiver connected to sample said early bus clock signal and gated by said output of said first additional delay line,
a means for adjusting said first variable delay so said first clocked input receiver samples said early bus clock signal just as said early bus clock signal transitions,
a second clocked input receiver connected to sample said late bus clock signal and gated by said output of said second additional delay line,
a means for adjusting said second variable delay so said second clocked input receiver samples said late bus clock signal just as said late bus clock signal transitions,
whereby said output of said third additional delay line is synchronized to a time halfway between said outputs of said first and said second additional delay lines, and said output of said third additional delay line provides an internal device clock signal.

81. The bus subsystem with the features of embodiment 80 further comprising a semiconductor device having

a first one of said low-skew clock generator circuits which generates a "true" internal device clock signal and
a second one of said low-skew clock generator circuits connected to generate a "complement" internal device clock signal synchronized with but opposite in logical value to said "true" internal device clock signal.

82. A DRAM device designed to be connected to an external bus having a plurality of bus lines for carrying substantially all address, data and control information needed by said DRAM device as a sequential series of bits, said control information including device-select information, said external bus containing substantially fewer said bus lines than the number of bits in a single address, and said bus carrying device-select information without the need for separate device-select lines connected directly to said DRAM device, said DRAM device comprising

an array of memory cells connected in rows and columns, each of said memory cells adapted to store one of said bits,
a row address selection means for selecting one of said rows,
a column sense amp connected to each of said columns, each of said column sense amps adapted to latch one of said bits as a binary logical value or to precharge to a selected state,
a column decoding means connected to each of said column sense amps for selecting a plurality of said column sense amps for inputting one of said bits to or outputting one of said bits from said memory cells,
an internal I/O bus having a plurality of internal I/O lines wherein each of said internal I/O lines is connected to a plurality of said column sense amps, and
a plurality of bus connection means designed to connect said internal I/O lines to said external bus,
whereby a selected bit of said sequential series of bits can be transferred from said external bus to a selected one of said memory cells or said bit contained in a selected one of said memory cells can be transferred to said external bus.

83. The DRAM device with the features of embodiment 82 further comprising

an output driver connected to one said bus connection means,
an output multiplexer having an output connected to said output driver and a plurality of inputs, each of said inputs being connected to one of said internal I/O lines, and
a control means to select whether said output driver can drive said external bus,
whereby a plurality of memory cells are selected using said row address selection means and said column decoding means and a plurality of bits contained in said plurality of memory cells are output through said column sense amps to said internal I/O bus to said output multiplexer to said output driver to said external bus.

84. The DRAM device with the features of embodiment 82 further comprising

a plurality of input receivers connected to one of said bus data lines and to said internal I/O bus,
a selection means for selecting said input receivers one by one to sense and store, one at a time, the bits of said sequential series of bits, and
a control means to select whether an input receiver can drive said internal I/O bus, whereby a bit of said sequential series of bits is input from said external bus through one of said input receivers to one of said internal I/O lines to one of said column sense amps to one of said memory cells.

85. The DRAM device with the features of embodiment 82 further comprising

a first and a second half-array of said memory cells wherein each said row of said array of said memory cells is subdivided into two parts,
a first and a second one of said internal I/O buses connected to said column sense amps in said first and said second half-arrays, respectively, and
a column decoder means to gate selected ones of said column sense amps connected to said memory cells in a selected row of said first and said second half-arrays simultaneously.

86. The DRAM device with the features of embodiment 85 wherein said column decoder means selects sixteen column sense amps at a time.

87. The DRAM device with the features of embodiment 82 wherein said external bus operates at a certain speed and wherein said DRAM device includes four of said internal I/O buses, each of which operates at one-fourth the speed of said external bus.

88. The DRAM device with the features of embodiment 82 further comprising

a means for precharging one of said column sense amps to a precharged state from which a binary logical

value can quickly be loaded into said column sense amp,
if said column sense amp contains a binary logical value, a means for latching the logical value currently contained in said column sense amp and
a means for instructing said DRAM device to precharge said column sense amp or latch said binary logical value in said column sense amp.

89. The DRAM device with the features of embodiment 88 further comprising a means for instructing said DRAM device to precharge said column sense amp without further instruction whenever said row address selection means selects a different one of said rows.

90. The DRAM device with the features of embodiment 88 further comprising a means for instructing said DRAM device to precharge said column sense amp without further instruction at a first or a second preselected time after latching the latest said binary logical value, said first preselected time being long enough for said DRAM to latch said binary logical value into said column sense amp and transfer said binary logical value into memory or onto one of said internal I/O lines, and said second preselected time being a variable which can be stored in said DRAM device whereby said DRAM can latch a binary logical value into said column sense amp for transferring said binary logical value into or out of a selected said memory cell, then precharge to allow a faster subsequent read or write.

91. A package containing

a semiconductor die having a side, circuitry and a plurality of connecting areas positioned along or near said side, spaced at a selected pitch and connected to said circuitry,
said package comprising a plurality of bus connecting means for connecting to a plurality of external bus lines, each of said external bus lines corresponding to one of said connecting areas, each of said bus connecting means being

positioned on a first side of said package,
connected to one said external bus line and to

said corresponding connecting area on said semiconductor die, and

spaced at a pitch substantially identical to said selected pitch of said connecting areas,

whereby each of said external bus lines can be connected to said corresponding connecting area on said semiconductor die by bus connection means positioned along a single side of said package.

92. The package with the features of embodiment 91 further comprising a plurality of said bus connecting means wherein each of said bus connecting means includes

a pin adapted for connection to one of said external bus lines and
a wire connecting said pin to one of said connecting areas on said semiconductor die,
said wire having an effective lead length less than about 4 millimeters and wherein the effective lead length of said wire of each of said bus connection means for said package is approximately equal.

93. A plurality of packages with the features of embodiment 91 wherein at least two of said semiconductor die are memory devices, each of said packages being generally flat, having a top and a bottom, and wherein
said packages are physically secured adjacent and parallel to each other in a stack,
where a first one of said packages is adjacent to a second one of said packages in said stack, said top of said first package is substantially aligned with said bottom of said second package, and
said bus connecting means of each of said packages are substantially aligned and are lying substantially in a plane.

94. The plurality of packages with the features of embodiment 93 further comprising a plurality of stacks wherein each of said bus connecting means can be electrically connected to corresponding said bus connecting means in each of said stacks.

95. A semiconductor device capable of use in a semiconductor bus architecture including a plurality of semiconductor devices connected in parallel to a bus wherein said bus includes a plurality of bus lines for carrying substantially all

address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said bus, and has substantially fewer bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said bus, and
at least one modifiable identification register accessible to said bus through said connection means, whereby data may be transmitted to said register via said bus and enable said device thereafter to be uniquely identified.

96. The semiconductor device with the features of embodiment 95 wherein said semiconductor device is a memory device which connects substantially only to said bus and sends and receives substantially all address, data and control information over said bus.

97. A semiconductor device capable of use in a semiconductor bus architecture including a plurality of semiconductor devices connected in parallel to a bus wherein said bus includes a plurality of bus lines for carrying substantially all address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said bus, and has substantially fever bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said bus, and
at least one modifiable register to hold device address information, said modifiable register accessible to said bus through said connection means, whereby data may be transmitted to said register via said bus which enables said device thereafter to respond to a predetermined range of addresses.

98. The semiconductor device with the features of embodiment 97 wherein said semiconductor device is a memory device which connects substantially only to said bus and sends and receives substantially all address, data and control information over said bus.

99. The semiconductor device with the features of embodiment 98 wherein said memory device has at least one discrete memory section and also has at least one modifiable address register adapted to store memory address information which corresponds to each said discrete memory section.

100. The semiconductor device with the features of embodiment 99 wherein said memory address information comprises a pointer to said discrete memory section.

101. The semiconductor device with the features of embodiment 100 wherein said discrete memory section has a top and a bottom and said memory address information comprises pointers to said top and said bottom.

102. The semiconductor device with the features of embodiment 100 wherein said memory address information comprises

a pointer to said discrete memory section and
a range value indicating the size of said discrete memory section.

103. A semiconductor device capable of use in a semiconductor bus architecture including a plurality of semiconductor devices connected in parallel to a bus wherein said bus includes a plurality of bus lines for carrying substantially all address, data and control information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said bus, and has substantially fewer bus lines than the number of bits in a single address, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said bus, and
at least one modifiable access-time register accessible to said bus through said connection means, whereby data may be transmitted to said register via said bus which establishes a predetermined amount of time that said semiconductor device thereafter must wait before using said bus in response to a request.

104. The semiconductor device with the features of embodiment 103 wherein said semiconductor device is a memory device which connects substantially only to said bus and sends and receives substantially all address, data and control information over said bus.

105. The semiconductor device with the features of embodiment 103 further comprising at least two access-time registers and one of said access-time registers is permanently programmed to contain a fixed value and at least one of said access-time registers can be modified by information carried on said bus.

106. A semiconductor device capable of use in a semiconductor bus architecture including a plurality of semiconductor devices connected in parallel to a bus wherein said bus includes a plurality of bus lines for carrying substantially all address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said bus, and has substantially fewer bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, and wherein each said bus line is a terminated transmission line, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said bus, and
a bus line driver capable of producing a low-voltage-swing signal on one of said terminated transmission lines.

107. The semiconductor device with the features of embodiment 103 wherein said semiconductor device is a memory device which connects substantially only to said bus and sends and receives substantially all address, data and control information over said bus.

108. A semiconductor device capable of use in a semiconductor bus architecture including a plurality of semiconductor devices connected in parallel to a bus wherein said bus includes a plurality of bus lines for carrying substantially all address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said bus, and has substantially fewer bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, said bus further including at least one bus clock line for carrying early and late bus clock signals, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said bus, and
an internal device clock generating means which generates an internal device clock synchronized to a time halfway between said early and said late bus clock signals.

109. The semiconductor device with the features of embodiment 108 wherein said bus further includes a first and a second one of said bus clock lines, said first bus clock line carries said early bus clock signal and said second bus clock line carries said late bus clock signal, said semiconductor device further comprising a means to detect said early bus clock signal on said first bus clock line and a means to detect said late bus clock signal on said second bus clock line.

110. The semiconductor device with the features of embodiment 109 wherein said semiconductor device is a memory device which connects substantially only to said bus and sends and receives substantially all address, data and control information over said bus.

111. A semiconductor device capable of use in a semiconductor bus architecture including a plurality of semiconductor devices connected in parallel to a bus wherein said bus includes a plurality of bus lines for carrying as a sequential series of bits substantially all address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said bus, and has substantially fewer bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said bus,
a plurality of input receivers connected to one of said bus data lines and
a selection means for selecting said input receivers one by one to sense and store, one at a time, the bits of said sequential series of bits.

112. The semiconductor device with the features of embodiment 111 wherein said semiconductor device is a memory device which connects substantially only to said bus and sends and receives substantially all address, data and control information over said bus.

113. The semiconductor device with the features of embodiment 112 wherein two input receivers are connected to one of said bus lines.

114. A semiconductor device capable of use in an architecture for a semiconductor system bus including a plurality of semiconductor devices connected in parallel to a bus wherein said bus system includes a plurality of bus lines for carrying substantially all address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said system bus, and has substantially fewer bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said system bus,
an internal input/output bus within said semiconductor device having more lines than said system bus, and
a means for multiplexing the lines of said internal bus to the lines of said system bus, whereby said system bus can run at a higher speed than said internal bus.

115. The semiconductor device with the features of embodiment 114 wherein said semiconductor device is a memory device which connects substantially only to said system bus and sends and receives substantially all address, data and control information over said system bus.

116. A semiconductor device capable of use in an architecture for a semiconductor system bus including a plurality of semiconductor devices connected in parallel to a bus wherein said system bus includes a plurality of bus lines for carrying substantially all address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said system bus, and has substantially fewer bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said system bus,
an internal input/output bus within said semiconductor device having more lines than said system bus,
a means for multiplexing the lines of said internal bus to the lines of said system bus, whereby said system bus can run at a higher speed than said internal bus, and
at least one modifiable identification register accessible to said system bus through said connection means, whereby data may be transmitted to said register via said system bus and which enables said device thereafter to be uniquely identified.

117. The semiconductor device with the features of embodiment 116 wherein said semiconductor device is a memory device which connects substantially only to said system bus and sends and receives substantially all address, data and control information over said system bus.

118. A semiconductor device capable of use in an architecture for a semiconductor system bus including a plurality of semiconductor devices connected in parallel to a bus wherein said system bus includes a plurality of bus lines for carrying substantially all address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said system bus, and has substantially fewer bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said system bus,
an internal input/output bus within said semiconductor device having more lines than said system bus,
a means for multiplexing the lines of said internal bus to the lines of said system bus, whereby said system bus can run at a higher speed than said internal bus, and
at least one modifiable register to hold device address information, said modifiable register accessible to said system bus through said connection means, whereby data may be transmitted to said register via said system

bus which enables said device thereafter to respond to a predetermined range of addresses.

119. The semiconductor device with the features of embodiment 118 wherein said semiconductor device is a memory device which connects substantially only to said system bus and sends and receives substantially all address, data and control information over said system bus.

120. The semiconductor device with the features of embodiment 119 wherein said memory device has at least one discrete memory section and also has at least one modifiable address register adapted to store memory address information which corresponds to each said discrete memory section.

121. A semiconductor device capable of use in an architecture for a semiconductor system bus including a plurality of semiconductor devices connected in parallel to a bus wherein said system bus includes a plurality of bus lines for carrying substantially all address, data and control information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said system bus, and has substantially fewer bus lines than the number of bits in a single address, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said system bus,
an internal input/output bus within said semiconductor device having more lines than said system bus,
a means for multiplexing the lines of said internal bus to the lines of said system bus, whereby said system bus can run at a higher speed than said internal bus, and
at least one modifiable access-time register accessible to said system bus through said connection means, whereby data may be transmitted to said register via said system bus which establishes a predetermined amount of time that said semiconductor device thereafter must wait before using said system bus in response to a request.

122. The semiconductor device with the features of embodiment 121 wherein said semiconductor device is a memory device which connects substantially only to said system bus and sends and receives substantially all address, data and control information over said system bus.

123. The semiconductor device with the features of embodiment 121 further comprising at least two access-time registers and one of said access-time registers is permanently programmed to contain a fixed value and at least one of said access-time registers can be modified by information carried on said system bus.

124. A semiconductor device capable of use in a semiconductor bus architecture including a plurality of semiconductor devices connected in parallel to a bus wherein said bus includes a plurality of bus lines for carrying substantially all address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said bus, and has substantially fewer bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, wherein said address, data, control and device-select information is carried over said bus in the form of request packets and bus transactions, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said bus,
a means to receive said request packets over said bus,
a means to decode information in said request packets, and
a means to respond to said information in said request packets.

125. The semiconductor device with the features of embodiment 124 wherein said means to decode information in said request packet further comprises

a means to identify and decode said control information in said request packet,
a means to identify and decode said device-select information in said request packet,
a means to identify and decode said address information in said request packet and
a means to determine whether said control information or said address information instructs said semiconductor device to begin a response.

126. The semiconductor device with the features of embodiment 124 wherein each of said bus transactions is carried out in response to said address and said control information in one of said request packets, and wherein said

means to identify and decode information in said request packets includes a means to identify a sequence of bytes on said bus as one of said request packets containing said address and said control information, said control information including information about the type of said bus transaction being requested and the access time which needs to intervene before beginning said bus transaction over said bus and said address and said control information includes device-select information instructing one or more said semiconductor devices to respond to said address and said control information.

127. The semiconductor device with the features of embodiment 124 further comprising

a plurality of sense amplifiers adapted to precharge to a selected state or to latch a bit of information,
a means to hold said sense amplifiers in an unmodified state after latching one of said bits of information,
a means to precharge said sense amplifiers and
a means for selecting whether said semiconductor device should precharge said sense amplifiers or should hold said sense amplifiers in an unmodified state.

128. The semiconductor device with the features of embodiment 124 wherein said means to respond to said information, where said information is control information, further comprises a means to

transfer a data block during a data block transfer, further including a means to

read data from said semiconductor device and
write data into said semiconductor device, and

initiate a data block transfer,
transfer a data block of a selected size,
transfer a data block at a selected time,
access a control register, including a means to read from or write to said control register, or
select normal or page-mode access.

129. The semiconductor device with the features of embodiment 124 further comprising a means to respond to said information in said request packet if said information includes a device identification number unique to said semiconductor device.

130. The semiconductor device with the features of embodiment 124 further comprising a means to respond to said information in said request packet if said information includes a special device identification number which calls for said semiconductor device to respond.

131. The semiconductor device with the features of embodiment 124 further comprising a means to respond to said information in said request packet if said information includes an address unique to said semiconductor device.

132. The semiconductor device with the features of embodiment 124 further comprising a means to interpret said control information and decode the time to wait before beginning said bus transaction over said bus.

133. The semiconductor device with the features of embodiment 124 further comprising a means to interpret said control information and decode the size of a data block to transfer during one of said bus transactions.

134. The semiconductor device with the features of embodiment 124, 125, 126, 127, 128, 129, 130, 131, 132 or 133 wherein said semiconductor device is a memory device which connects substantially only to said bus and sends and receives substantially all address, data and control information over said bus.

135. A semiconductor device capable of use in a semiconductor bus architecture including a plurality of semiconductor devices connected in parallel to a bus wherein said bus includes a plurality of bus lines for carrying substantially all address, data, control and device-select information needed by said semiconductor device for communication with substantially every other semiconductor device connected to said bus, and has substantially fewer bus lines than the number of bits in a single address, and carries device-select information for said semiconductor device without the need for a separate device-select line connected directly to said individual semiconductor device, wherein said address, data, control and device-select information is carried over said bus in the form of request packets and bus transactions, said semiconductor device comprising

connection means adapted to connect said semiconductor device to said bus,
a means to encode address and control information in said request packets and
a means to send said request packets over said bus.

136. The semiconductor device with the features of embodiment 135 further comprising a means to request a bus transaction wherein each of said bus transactions is carried out in response to said address and said control information in one of said request packets, and wherein said means to encode information in said request packets includes a means to mark a sequence of bytes on said bus as one of said request packets, said control information including information about the type of said bus transaction being requested and the access time which needs to intervene before beginning said bus transaction over said bus and said address and said control information includes device-select information instructing one or more said semiconductor devices to respond to said address and said control information.

137. The semiconductor device with the features of embodiment 135 wherein one or more of said plurality of semiconductor devices has a unique device identification number, said semiconductor device further comprising a means to send control information to a specific one of said plurality of semiconductor devices by including in said request packet a selected said device identification number.

138. The semiconductor device with the features of embodiment 135 wherein each of said plurality of semiconductor devices is adapted to respond to a special device identification number, said semiconductor device further comprising a means to send control information to each of said plurality of semiconductor devices by including in said request packet said special device identification number.

139. The semiconductor device with the features of embodiment 135 wherein one or more of said plurality of semiconductor devices is a memory device having a plurality of addresses, said semiconductor device further comprising a means to send control information to a specific address or range of addresses in one of said plurality of semiconductor devices by including said specific address or range of addresses in said request packet.

140. The semiconductor device with the features of embodiment 135 wherein at least one of said request packets is a request packet requesting a bus transaction which is followed by a corresponding one of said bus transactions, said semiconductor device further comprising a means to encode said control information to specify directly or indirectly the time between the end of said request packet requesting a bus transaction and said corresponding bus transaction over said bus.

141. The semiconductor device with the features of embodiment 140 wherein one type of said bus transactions is a transfer of a data block, said semiconductor device further comprising a means to encode said control information to specify the size of said data block to transfer.

142. The semiconductor device with the features of embodiment 140 further comprising a means to keep track of current and pending bus transactions, whereby collisions on said bus are avoided because said semiconductor device avoids initiating bus transactions which would conflict with current or pending bus transactions.

143. The semiconductor device with the features of embodiment 135 wherein said semiconductor device is a first master device and one of said plurality of semiconductor devices is a second master device, further comprising

a collision detecting means whereby said first master device when sending a first one of said request packets can detect said second master device sending a colliding one of said request packets, where said colliding request packet may be sent simultaneous with the initial sending of or overlapping the sending of said first request packet, and
an arbitration means whereby said first and said second master devices select a priority order in which each of said master devices will be allowed to access said bus sequentially.

144. The semiconductor device with the features of embodiment 143 wherein said semiconductor device is a master device and at least one of said plurality of semiconductor devices is a master device, each of said master devices has a master ID number and each of said request packets includes a master ID position which is a predetermined number of bits in a predetermined position in said request packet, and wherein said collision detection means comprises

a means for said semiconductor device to send its master ID number in said request packet and

a means to detect a collision and invoke said arbitration means if said semiconductor device detects any other master ID number in said master ID position.

145.The semiconductor device with the features of embodiment 144 wherein said system bus architecture includes a means for carrying information on said bus during bus cycles, said semiconductor device further comprising

a means for driving a selected bus line or lines during at least one selected bus cycle while sending each said request packet,

a means for monitoring said selected bus line or lines to see if another said master device is sending one of said colliding request packets and

a means for informing all said master devices that a collision has occurred and for invoking said arbitration means.

146.The semiconductor device with the features of embodiment 145 further comprising

a means, when sending a request packet, for driving a selected bus line or lines with a certain current during at least one selected bus cycle,

a means for monitoring said selected bus line or lines for a greater than normal current to see if another said master device is driving that line or lines,

a means for detecting said greater than normal current, and

a means for informing all said master devices that a collision has occurred and for invoking said arbitration means.

147.The semiconductor device with the features of embodiment 143 wherein said arbitration means comprises

a means for initiating an arbitration cycle,

a means for allocating a single bus line to each said master device during at least one selected bus cycle relative to the start of said arbitration cycle,

a means for allocating each said master device to a single bus line during one of said selected bus cycles if there are more master devices than available bus lines,

a means for each of said master devices which sent one of said colllding request packets to drive said bus line allocated to said master device during said selected bus cycle, and

a means in at least one of said master devices for storing information about which master devices sent one of said colliding request packets,

whereby said master devices can monitor selected bus lines during said arbitration cycle and identify each said master device which sent one of said colliding request packets.

148.The semiconductor device with the features of embodiment 143 wherein said arbitration means comprises

a means for identifying each of said master devices which sent one of said colliding request packets,

a means for assigning a priority to each said master device which sent one of said colliding request packets, and

a means for allowing each said master device which sent one of said colliding request packets to access the bus sequentially according to that priority.

149.The semiconductor device with the features of embodiment 143 wherein said priority is based on the physical location of each of said master devices.

150.The semiconductor device with the features of embodiment 143 wherein said priority is based on said master ID number of said master devices.

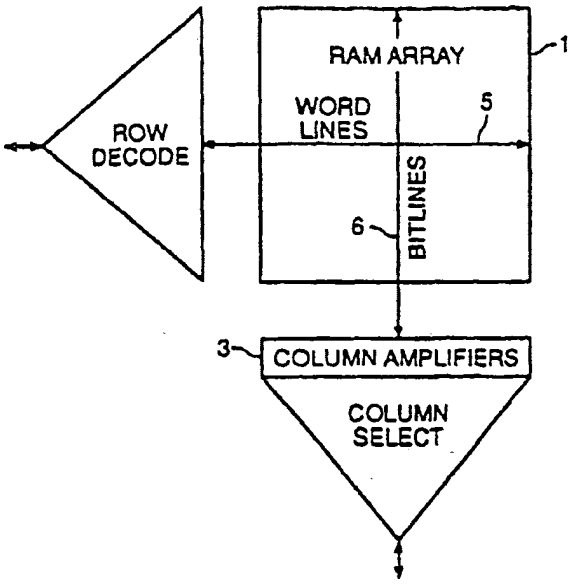
RAM ARRAY
1
WORD LINES
5
ROW DECODE
BITLINES
6
3
COLUMN AMPLIFIERS
COLUMN SELECT

FIG. 1

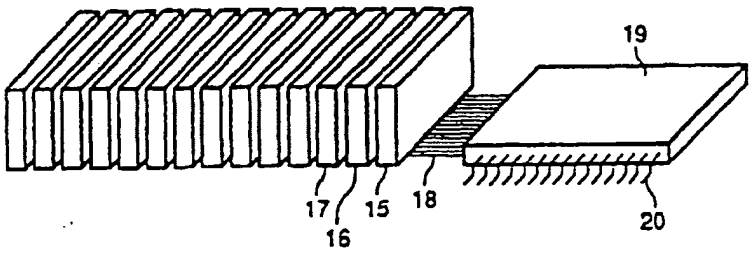
19
17
16
15
18
20

FIG. 3

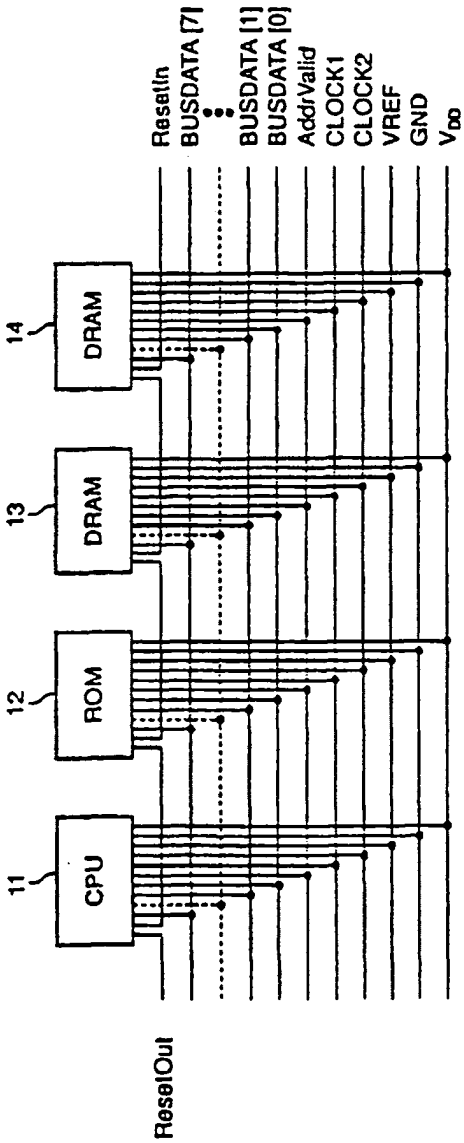
ResetIn
BUSDATA [7]
BUSDATA [1]
BUSDATA [0]
AddrValid
CLOCK1
CLOCK2
VREF
GND
V_{DD}
14
DRAM
13
DRAM
12
ROM
11
CPU
ResetOut

FIG. 2

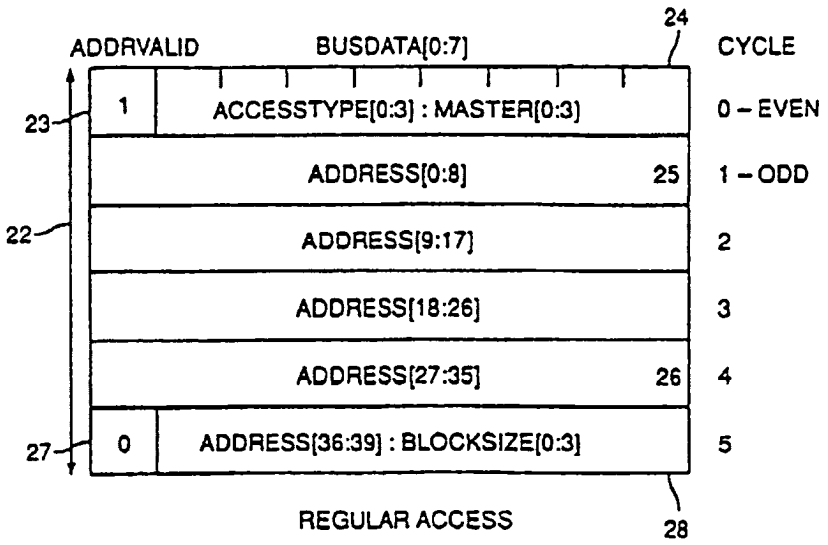
24
ADDRVALID
BUSDATA[0:7]
CYCLE
23
1
ACCESSTYPE[0:3] : MASTER[0:3]
0 – EVEN
ADDRESS[0:8]
25
1 – ODD
22
ADDRESS[9:17]
2
ADDRESS[18:26]
3
ADDRESS[27:35]
26
4
27
0
ADDRESS[36:39] : BLOCKSIZE[0:3]
5
REGULAR ACCESS
28

FIG. 4

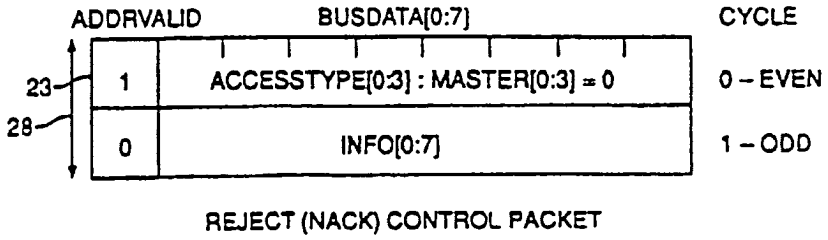
ADDRVALID
BUSDATA[0:7]
CYCLE
23
1
ACCESSTYPE[0:3] : MASTER[0:3] = 0
0 – EVEN
28
0
INFO[0:7]
1 – ODD
REJECT (NACK) CONTROL PACKET

FIG. 5

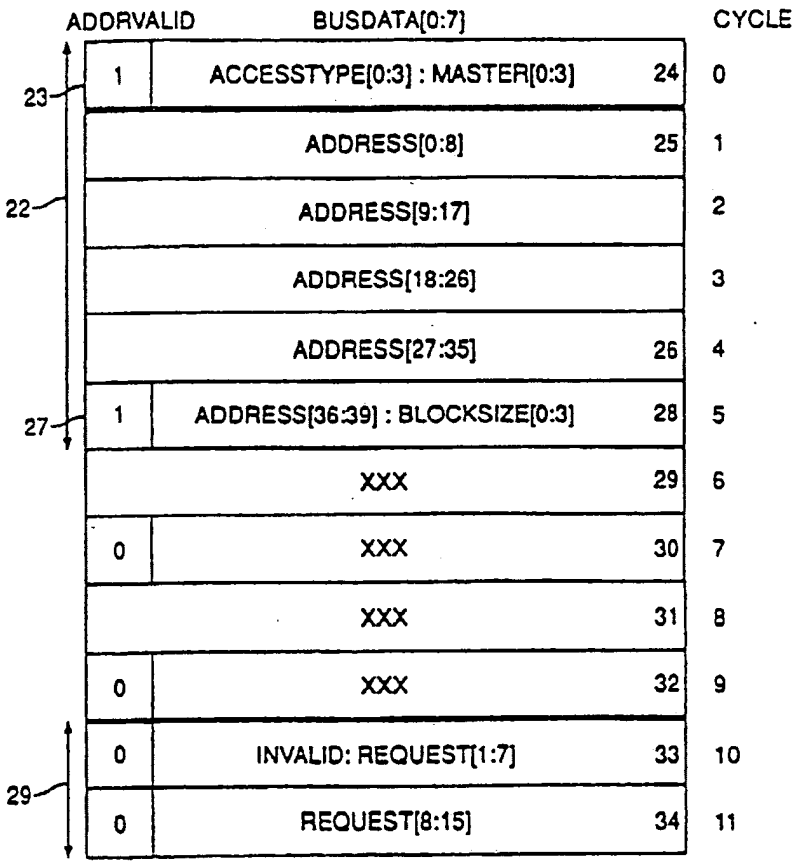
ADDRVALID
BUSDATA[0:7]
CYCLE
1
ACCESSTYPE[0:3] : MASTER[0:3]
24
0
23
ADDRESS[0:8]
25
1
22
ADDRESS[9:17]
2
ADDRESS[18:26]
3
ADDRESS[27:35]
26
4
1
ADDRESS[36:39] : BLOCKSIZE[0:3]
28
5
27
XXX
29
6
0
XXX
30
7
XXX
31
8
0
XXX
32
9
0
INVALID: REQUEST[1:7]
33
10
29
0
REQUEST[8:15]
34
11

FIG. 6


41
42
43
A
B
44
45
a
b

FIG. 7a

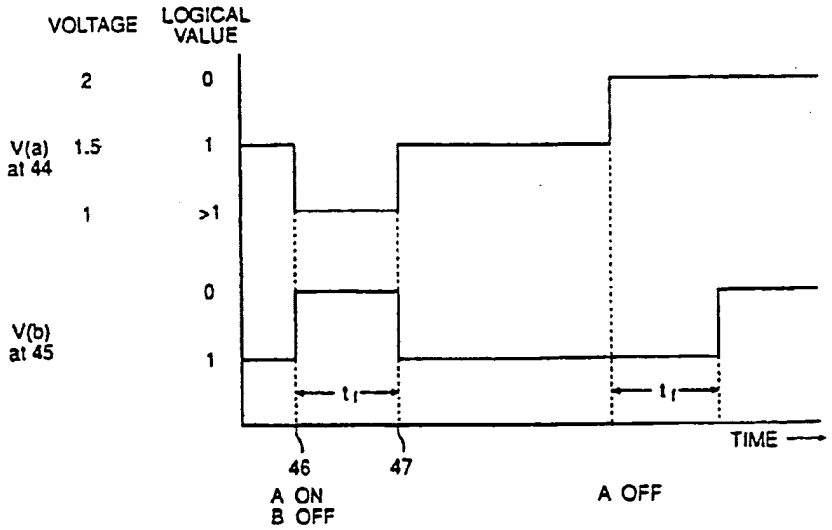
VOLTAGE
LOGICAL VALUE
2
0
V(a) at 44
1.5
1
1
>1
0
V(b) at 45
1
t_f
t_f
TIME
46
47
A ON
B OFF
A OFF

FIG. 7b

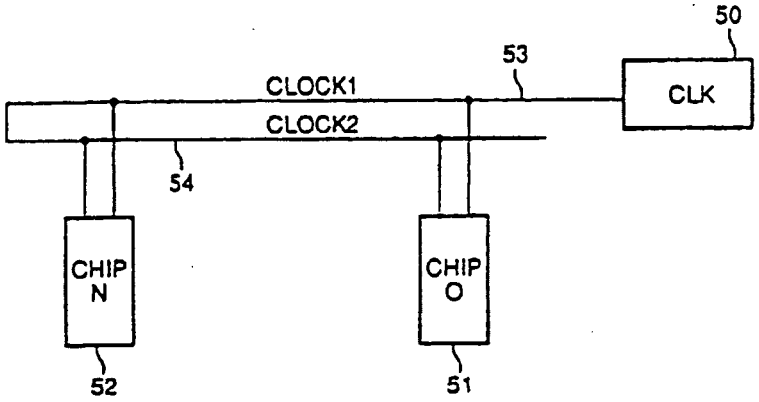
50
53
CLOCK1
CLOCK2
CLK
54
CHIP N
CHIP O
52
51

FIG. 8a

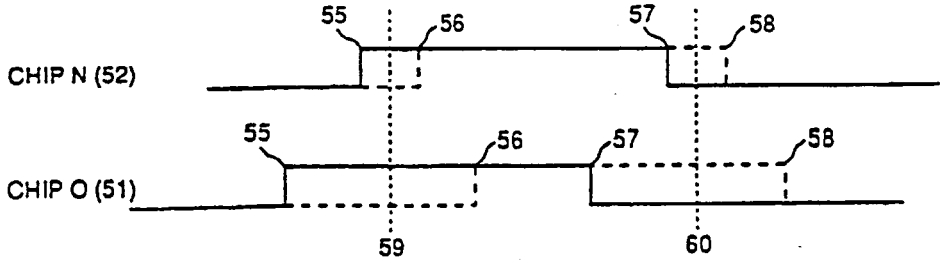
55
56
57
58
CHIP N (52)
55
56
57
58
CHIP O (51)
59
60

FIG. 8b

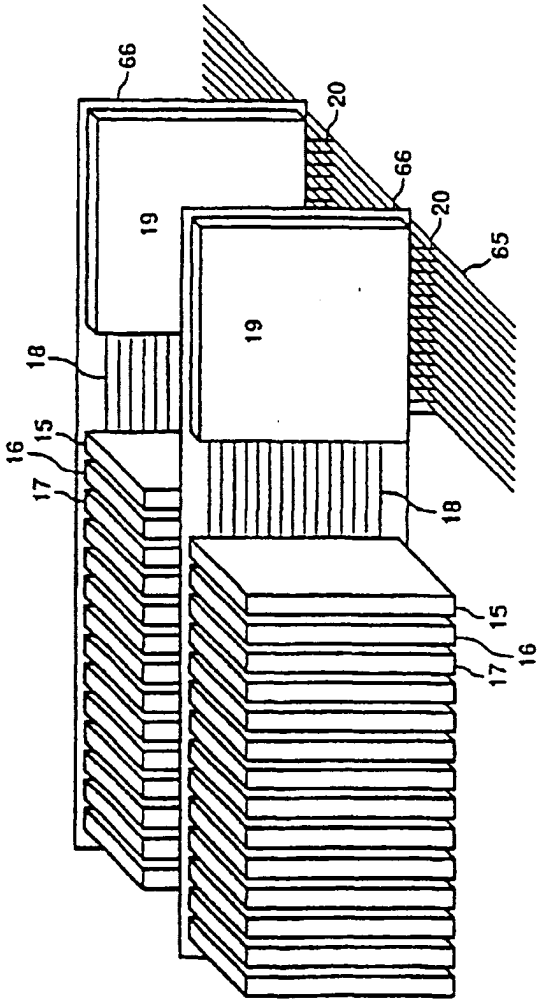

FIG. 9

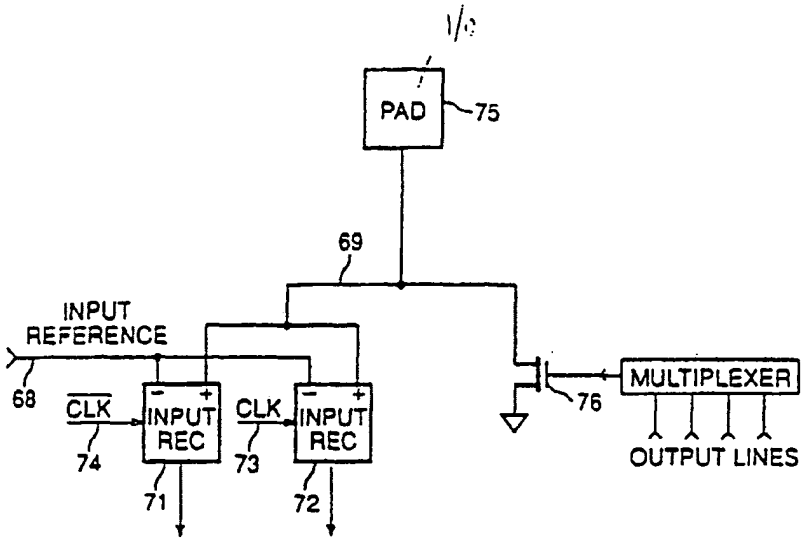
PAD
75
69
INPUT
REFERENCE
68
CLK
74
INPUT
REC
71
CLK
73
INPUT
REC
72
76
MULTIPLEXER
OUTPUT LINES

FIG. 10

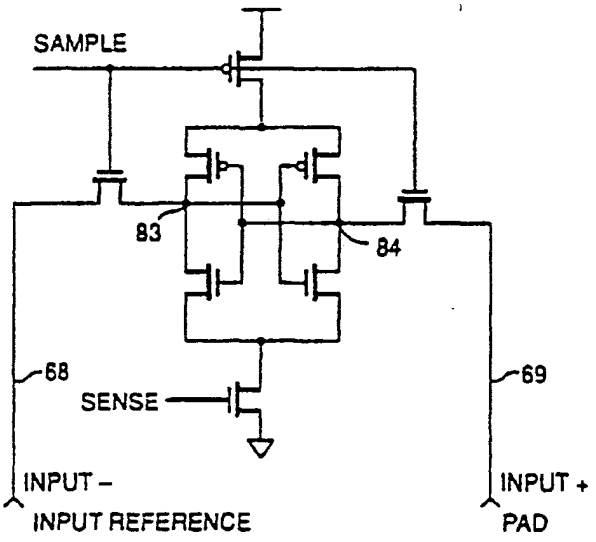
SAMPLE
83
84
68
SENSE
69
INPUT –
INPUT REFERENCE
INPUT +
PAD

FIG. 11

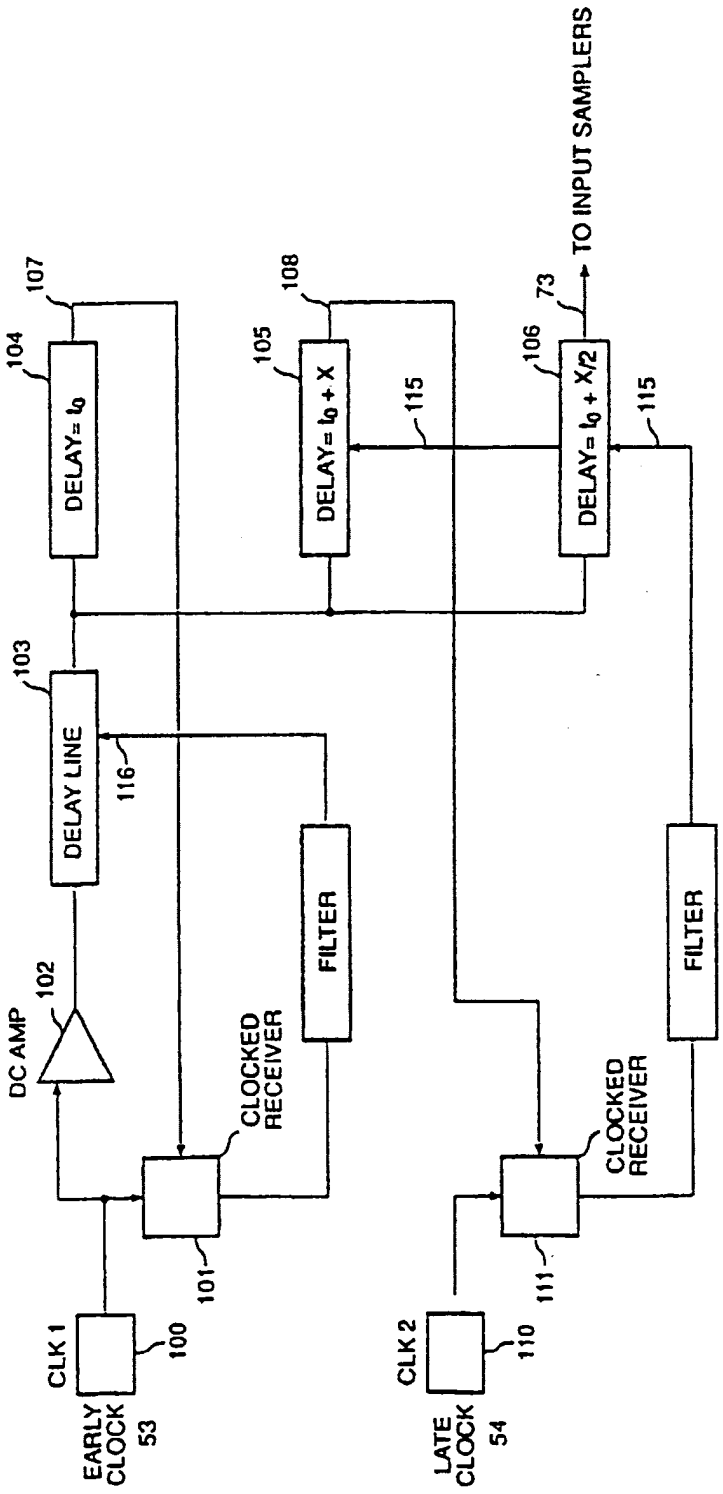
TO INPUT SAMPLERS
73
107
108
104
105
106
DELAY= t0
DELAY= t0 + X
DELAY= t0 + X/2
115
115
103
DELAY LINE
116
FILTER
FILTER
DC AMP
102
CLOCKED RECEIVER
CLOCKED RECEIVER
101
111
CLK 1
100
CLK 2
110
EARLY CLOCK
53
LATE CLOCK
54

FIG. 12

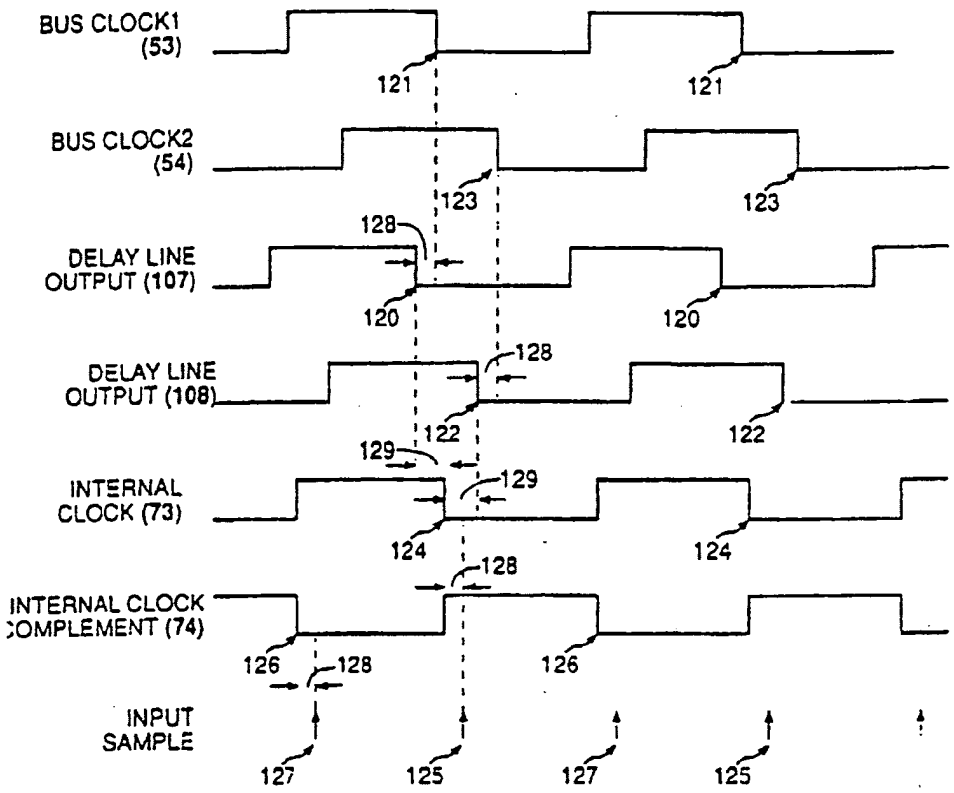
BUS CLOCK1 (53)
121
121
BUS CLOCK2 (54)
123
123
128
DELAY LINE OUTPUT (107)
120
120
128
DELAY LINE OUTPUT (108)
122
122
129
129
INTERNAL CLOCK (73)
124
124
128
INTERNAL CLOCK COMPLEMENT (74)
126
128
126
INPUT SAMPLE
127
125
127
125

FIG. 13

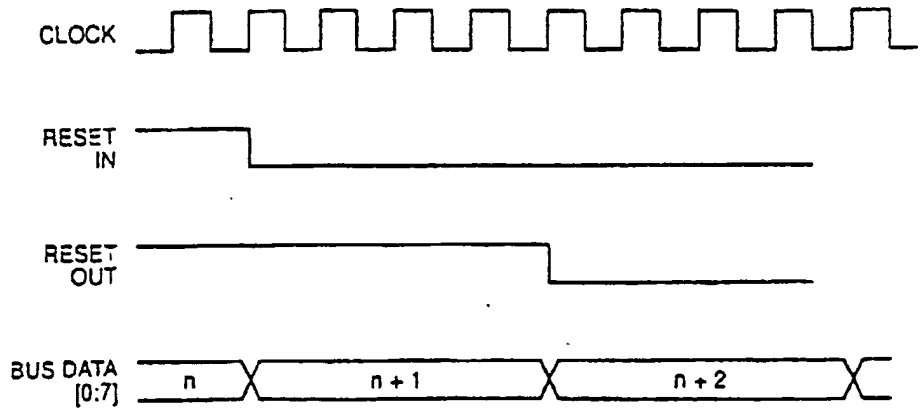
CLOCK
RESET IN
RESET OUT
BUS DATA [0:7]
n
n + 1
n + 2

FIG. 14

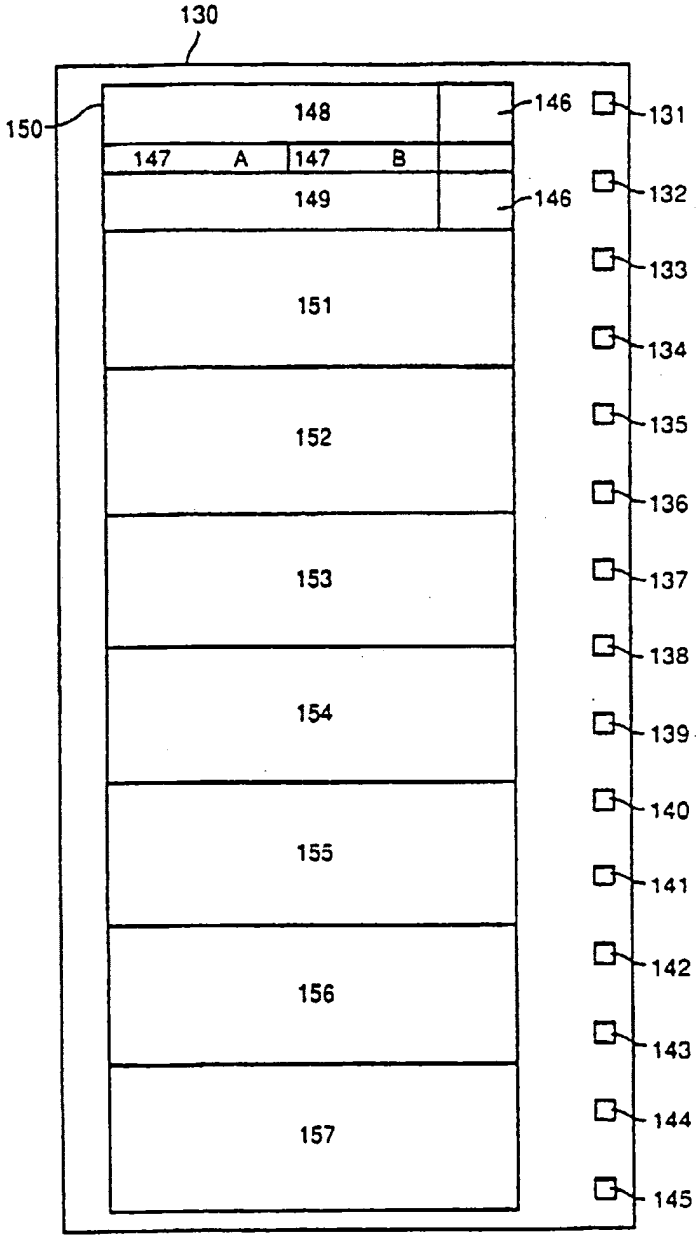
130
150
148
146
131
147
A
147
B
149
146
132
151
133
134
152
135
136
153
137
138
154
139
140
155
141
142
156
143
144
157
145

FIG. 15

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 00 10 0018

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
A	US 4 630 193 A (KRIS BRYAN) 16 December 1986 (1986-12-16) * column 1, line 58 - column 2, line 4 * * column 2, line 40 - column 5, line 33 * * abstract; figure 1 *	1-28	G06F1/04 G06F1/12 G06F13/00 G06F13/16 G06F13/36 G06F13/38
A	US 4 205 373 A (SHAH NIRANJAN S ET AL) 27 May 1980 (1980-05-27) * the whole document *	1-28	G11C7/00 G11C8/00 G11C8/04
A	US 4 710 904 A (SUZUKI ATSUSHI) 1 December 1987 (1987-12-01) * abstract *	1-28	

TECHNICAL FIELDS SEARCHED (Int.Cl.7)

G06F
G11C

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	26 May 2000	Nguyen Xuan Hiep, C

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 00 10 0018

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

26-05-2000

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 4630193 A	16-12-1986	WO 8203931 A	11-11-1982
		EP 0077328 A	27-04-1983
US 4205373 A	27-05-1980	DE 2920490 A	29-11-1979
		FR 2426937 A	21-12-1979
		GB 2021824 A,B	05-12-1979
		JP 1443561 C	08-06-1988
		JP 54152928 A	01-12-1979
		JP 62050862 B	27-10-1987
US 4710904 A	01-12-1987	JP 1765685 C	11-06-1993
		JP 4053035 B	25-08-1992
		JP 61054098 A	18-03-1986
		DE 3586810 A	17-12-1992
		DE 3586810 T	25-03-1993
		EP 0176226 A	02-04-1986
		KR 9102498 B	23-04-1991

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

EPA/EPO/OEB
D-80298 München
+49 89 2399 - 0
FAX +49 89 2399 - 4465

Ei
Pa

Ge

ter Meer, Steinmeister & Partner GbR
Einspruch gegen EP 1 197 830
Hynix Semiconductor ./. Rambus Inc.
zu Anlage U7

Eisenführ, Speiser & Partner
Patentanwälte Rechtsanwälte
Postfach 10 60 78
28060 Bremen
ALLEMAGNE

Formalities Officer
Name: Schall
Tel.: 2647
p.p. R. THOMAS

Date
06-09-2006

Reference	Application No./Patent No.
R 942	00100018.1 - 2201
Applicant/Proprietor: RAMBUS INC.	

Summons to attend oral proceedings pursuant to Rule 71(1) EPC

You are hereby summoned to attend oral proceedings arranged in connection with the above-mentioned European patent application.

The matters to be discussed are set out in the communication accompanying this summons (EPO Form 2906).

The oral proceedings, which will not be public, will take place before the examining division

on 30.11.06 at 13.00 hrs at the EPO,
PschorrHöfe, Bayerstr. 34, D-80335 München

No changes to the date of the oral proceedings can be made, except on serious grounds (see OJ EPO 10/2000, 456).
If you do not appear as summoned, the oral proceedings may continue without you (R. 71(2) EPC).
Your attention is drawn to Rule 2 EPC, regarding the language of the oral proceedings, and to the OJ EPO 9/1991, 489, concerning the filing of authorisations for company employees and lawyers acting as representatives before the EPO.

The final date for making written submissions and/or amendments (Rule 71a EPC), is 30.10.06.

The actual room number as well as the waiting room numbers will be given to you by the porter in the foyer at the above EPO address.

Parking is available free of charge in the underground car park. However, this applies only in the case of accessing the car park via the entrance "Zollstrasse".

1st Examiner:	2nd Member:	Chairman:	Legally qualified member:
Van Voorst Tot Voors	Prins L	Semple M	Cramer R

For the Examining Division

Annexes:
Confirmation of receipt (Form 2936)
Communication (EPO Form 2906)

Europäisches Patentamt · European Patent Office · Office européen des brevets

Registered letter with advice of delivery

to EPO postal service: 01.09.06

The examination is being carried out on the **following application documents**:

Description, Pages

1-124 as originally filed

Claims, Numbers

9-18	received on	27.08.2004	with letter of	26.08.2004
1-8	received on	04.11.2005	with letter of	03.11.2005

Drawings, Sheets

1/11-11/11 as originally filed

1). D4: "The CVAX CMTL - A CMOS Memory Controller Chip", Digital Technical Journal, N. 7, August 1988, pages 139 to 143, by D.K Morgan.

D5: "KA660 CPU Module Technical Manual" Digital Equipment Corporation, march 1990.

D6: Intel Application Note AP-132, "Designing Memory Systems with the 8K x 8 iRAM, June 1982, pages 3-41 to 3-45; Intel Preliminary Data Sheet "2186 - 8192x8 Bit Integrated RAM", September 1982, pages 3-281; and one undated sheet without page number relating to the 2186 family.

D7: "Memory Components Handbook" Intel 1985 (= D6 complete).

D4 and D6 have been introduced in the Appeal proceedings with respect of the specification EP-B-0525068 which was granted on the basis of the parent European patent application 91 908 374.1

D5 and D7 have been submitted by the Applicant during the oral proceedings held on 11-10-2005 in the divisional application EP 99 118 308.8.

2). The present divisional application 00 100 018.1 has been divided from the parent application 91 908 374.1. The parent application is based on the PCT application

As being indicated in the first official communication dated 16-10-2003, claim 1 of the present application filed with letter of 02-01-2000 did not fulfill the requirement of Article 76(1) in that a European divisional application may be filed only in respect of subject-matter which does not extend beyond the content of the earlier application as filed. It had been argued that the parent application 91 908 374.1 as originally filed consistently presents a narrow bus architecture as being an essential feature of the memory system (see also T0081/03): **the external bus including a plurality of bus lines for carrying substantially all address, data and control information needed by the semiconductor memory device connected to the external bus, wherein the external bus has substantially fewer bus lines than the number of bits in a single address.**

With replies dated 16-10-2003 and 04-11-2005 the Applicant filed two new sets of claims, each set introducing amendments to overcome the objection under Article 76(1) EPC.

In the Annex to the official communication Form 2081C dated 13-12-2005 the Applicant has been informed that following T166/84 the proceedings in the present case should be stayed until the outcome of case G01/05 is known, because amendments have been filed to overcome the Article 76(1) EPC objection.

On 25-01-2006 a request for legal opinion has been sent to the Directorate Patent Law, thereby referring to Applicant's reply dated 27-12-2005, which disagreed with the decision of the examining division to suspend the proceedings, and which

requested to proceed with the examination.

After receiving a legal advice from the Directorate Patent Law on 30-05-2006, the Examining Division has decided to withdraw the communication (EPO Form 2081C) of 13-12-2005 and to continue the examination in the present case. In view of the age of the present application, and because the Applicant in his letter dated 27-12-2005 has requested to proceed with the examination according to his prior request for accelerated examination, the Applicant has a legitimate interest in continuation. Furthermore, the Directorate Patent Law has indicated that there should be no automatism to stay proceedings in all affected pending proceedings,

It is noted that if during the further procedure in the present case the outcome of the case G1/05 becomes known, this will be applied immediately.

3). In the present case, only the second set of new claims is relevant (filed with letter 04-11-2005), as the Applicant obviously does not want to continue on the basis of the first set (filed with letter dated 16-10-2003).

4). Original claim 1 of the present application is not restricted to a specific narrow bus architecture, namely that the external bus includes a plurality of bus lines for carrying substantially all address, data and control information needed by the semiconductor memory device connected to the external bus, wherein the external bus has substantially fewer bus lines than the number of bits in a single address.

Only this specific narrow bus architecture has been originally disclosed in the description of the PCT application.

The subject-matter of original claim 1 thus does not comply with the requirements of the second sentence of Article 76(1) EPC

5). In present claim 1 the bus is defined narrower.

Present claim 1 now defines that a system comprises

a bus that includes a plurality of bus lines for carrying substantially all addresses, data and control information needed by each semiconductor coupled to the bus for communication with substantially every other semiconductor device connected to the bus, wherein the bus uses address multiplexing to convey a single memory address; and

a plurality of synchronic dynamic random access memory (DRAM) semiconductor devices coupled to the bus, each DRAM of the plurality of synchronous DRAM semiconductor devices including:

connection means adapted to connect the DRAM to the bus;
clock receiver circuitry (101, 111) for receiving a clock signal (53, 54);
a programmable access-time register for storing a value which is representative of a number of clock cycles of the clock signal (53, 54) to transpire after which the DRAM responds to a read request received synchronously with respect to the clock signal, the programmable access-time register being accessible to the bus through the connection means, wherein data is transmitted to the programmable access-time register over the bus to set the value in the programmable access-time register; and
a plurality of output drivers (76) for outputting data onto the bus (18, 65) in response to the read request, wherein the output drivers (76) output the data on the bus (18, 65) after the number of clock cycles of the clock signal transpire and synchronously with respect to the clock signal (53, 54), so that the read request and the corresponding response are separated by the number of clock cycles as selected by the value stored in the programmable access-time register, wherein each output driver of the plurality of output drivers (76) outputs the data onto the bus (18, 65) at a bus cycle data rate that is twice the data rate of the clock signal.

6). The passages of present claim 1 which have been indicated in bold have also been defined in the granted claim 1 of the European patent application 99 118 308.8, which is on its turn an application divided from the parent application.

Claim 1 of European patent application 99 118 308.8 is directed to one DRAM device

having at least one memory array, and defines in contrast to present claim 1 in fact only that the DRAM also comprises a plurality of input receivers to receive the read request synchronously with respect to the external clock signal, and that the external bus has substantially fewer bus lines than the number of bits in a single address.

With respect of the last mentioned feature it is noted that present claim 1 defines that the bus uses address multiplexing to convey a single memory address, so that in fact the same subject-matter is described in a different wording.

7). As a result, no relevant difference in subject-matter can be discovered between the first claims of both applications.

It is thus not clear which problem the present invention then seeks to solve.

The following has to be noted:

According to the EPO practice, a divisional application may not claim the same invention as the parent application or granted parent application, in order to avoid double patenting. In such a case the applicant has no legitimate interest in proceedings leading to the grant of a second patent for the same invention. However, this mechanism does not prevent the duplication when no objection of double patenting arises e.g. because the parent application has been amended during grant proceedings or it is refused, withdrawn or deemed to be withdrawn.

However, even if no objection of double patenting arises, there may be particular reasons why such practice which leads to a "duplication" is not justified, for instance, when the applicant withdraws the parent application in order to avoid the imminent refusal of the examining division and files a divisional application which is identical to the parent application. By doing this, the applicant can bring about a situation in which the same technical content has been discussed again, and keep his competitor in a position of legal uncertainty, which can be prolonged by a systematic filing of sequential divisional applications until the twenty-year patent term expires.

The present application has a first claim which is similar to claim 1 as granted in the

divisional application 99 118 308.8. Also in this case the objection of double patenting arises.

8). Apart from the last mentioned objection, Applicant's attention is asked for document D4 which has been introduced in the procedure of the other divisional application 99 118 308.8.

D4 appears to disclose a system in which a CPU is connected to a memory controller (CMCTL) by a synchronous CVAX bus. The memory controller is connected to up to four boards with each 16 MB DRAM memory through an asynchronous PMI bus. The CMCTL memory controller can be programmed to add slip cycles to memory read operations in increments of the CVAX bus cycle time. In order to read valid data from the DRAM memory, the CMCTL memory controller must wait a certain appropriate time until the DRAM output is stable. The only time reference the CMCTL controller has at its disposal to measure this time, is the clock signal on the CVAX bus. In order to deal with different configurations in terms of CVAX clock frequency used, the CMCTL must be programmable to wait more CVAX clock cycles if the CVAX clock frequency is high to ensure that the minimum time required for the DRAM memory to provide valid data, is adhered to. This can for instance be seen from table I of D4.

9). The subject-matter of present claim 1 differs from D4 in the following distinguishing features A, B, and C.

A: Dynamic Random Access Memory (DRAM) semiconductor devices are used, wherein the memory cells and all other circuits including the registers are integrated on a single chip;

B: the addresses must be multiplexed (meaning that the (external) bus has fewer bus lines than the number of bits in a single address);

C: each output driver of the plurality of output drivers outputs the data onto the (external) bus at a bus cycle data rate that is twice a rate of the (external) clock signal ("double data rate").

It is noted that the (external) clock signal refers to the bus clocks 53, 54 (see Figure 13), and that the output drivers output the data onto the (external) bus with the internal clock signal 73, 74 which is in synchronism with the (external) clock signals.

10). Given these differences over D4, the objective technical problem appears to be the provision of a system with memory devices which use the (external) bus in the most efficient way.

The present invention adds a synchronous interface to the DRAM devices in order to allow interleaving memory requests in a controlled manner thus providing a higher throughput. The present invention does not propose to integrate the memory controller with the DRAM memory.

D4 does not contain any teaching of how the efficiency of the external bus can be increased. Integrating the memory controller of D1 with the DRAM of the connected boards on a single device would certainly not lead to a more efficient use of the CVAX bus.

11). With respect to distinguishing feature A, the examining division considers that it is not obvious to combine the four memory modules shown in Figure 1 of D4 on a single chip, also in the light of D7. It is noted that D6 is taken from a larger document D7 and discussed in the decision of the Board of Appeal in the appeal proceedings for the parent application. From the complete document D7, one can clearly see that these pages taken from D7 were taken out of context. The selected pages contain a device labelled as "Synchronous 2167" in Figure 6. However, this use of the term "synchronous" has nothing to do with a synchronous bus protocol controlled by a clock signal as claimed.

D7 teaches that the refresh circuitry necessary for DRAM devices can be integrated on the DRAM device, instead of being taken care of centrally by the memory controller. The refreshing of the DRAM is an activity that can be delegated to individual DRAM devices, without surrending any form of central control of the memory controller, or inducing any loss of flexibility in terms of system configurations with different amounts of memory.

D7 provides two versions of DRAM devices with integrated refresh circuitry, also known as "iRAM". The "asynchronous" version performs refreshing autonomously. This can lead to high memory latency during refreshing. The "synchronous" version has a Refresh Enable input. This input can be used to control when the iRAM is allowed to perform refreshing. None of the disclosed iRAM devices has a synchronous bus interface in the sense of the present invention, with an external clock signal provided to the device.

D7 further teaches that the suggested integration of refresh circuitry into DRAM devices makes sense for small memories (64 kb), but a separate refresh controller is more cost effective in larger memories.

12). Furthermore, the present invention appears to be concerned with the throughput on the memory bus. To improve bus throughput, during the time that the memory is retrieving data, the memory bus could be used for something else. No conflicts must arise between the transfer of the retrieved data and any operation that was performed on the memory bus during the time that the DRAM was still retrieving that data.

The solution provided by the present invention appears to bring the synchronous domain to the DRAM device. This allows such interleaved bus accesses with well defined timing without conflicts as long as participants know which clock cycles they have at their disposal. the required features thus comprise an external clock signal provided to the DRAM device, and a register allowing the DRAM device to be programmed to provide the read data to the memory bus at a well defined moment in time not conflicting with any other transfers over the same memory bus.

Furthermore, in order for the skilled person to reach the claimed invention starting from D4, it appears that he would at least have to integrate the memory controller of D4 with the DRAM of D4 on a single device. Otherwise, no external clock signal would be provided to such a device as claimed.. However, even if the skilled person would do that, D4 is silent about when read data is provided on the CVAX bus. It only discloses that the memory controller can be made to wait a programmable number of clock cycles (for which the CVAX bus clock signal is used as a time reference) before

the data on the PMI bus is latched into the memory controller. It does not disclose that the read data is provided to the CVAX bus a predetermined number of clock cycles after the request. It appears to be more likely, in accordance with a commonly found scenario, that read data is collated and provided to the CVAX bus only some time later. Furthermore, buses like the CVAX bus are typically arbitrated, thus leading to a further delay, in contrast to the claimed invention in which the delay between request and response via the synchronous interface can precisely be controlled through the programmable access-time register.

13). Furthermore, it appears that with the synchronous interface and the programmable delay, the memory bus throughput can be increased despite the latency of individual memory requests. By interleaving memory reads, data can be made available for transfer via the synchronous interface at a rate independent of the latency of individual read requests. It becomes thus feasible to use both rising and falling edges of the clock signal to transfer data on the memory bus. Using "double data rate" on its own doesn't make any sense since the memory would not able to provide data with that rate. This shows that the claimed features differing from D4 are not to be assessed separately.

The differing features are considered not to be a mere aggregation of features which each provide their known effect without causing a combined effect.

14). Although the present application would be inventive with respect to the cited prior art, the claims are not allowable because no essential difference can be found with granted claim 1 of 99 118 308.8.

Richard van Voorst tot Voorst

(19) Europäisch Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 022 641 B1

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication and mention of the grant of the patent:
07.03.2007 Bulletin 2007/10

(21) Application number: **00100018.1**

(22) Date of filing: **16.04.1991**

(51) Int Cl.:
G06F 1/04 (2006.01) *G06F 1/12* (2006.01)
G06F 13/00 (2006.01) *G06F 13/16* (2006.01)
G06F 13/36 (2006.01) *G06F 13/38* (2006.01)
G11C 7/00 (2006.01) *G11C 8/00* (2006.01)
G11C 8/04 (2006.01)

(54) **System containing a plurality of DRAMS and a bus**

System mit einer Vielzahl von DRAMS und einem Bus

Système comportant multiple DRAMS et une bus

(84) Designated Contracting States:
DE FR GB IT

(30) Priority: **18.04.1990 US 510898**

(43) Date of publication of application:
26.07.2000 Bulletin 2000/30

(60) Divisional application:
05026720.2 / 1 640 847
06125946.1
06125954.5
06125958.6

(62) Document number(s) of the earlier application(s) in accordance with Art. 76 EPC:
91908374.1 / 0 525 068

(73) Proprietor: **RAMBUS INC.**
Los Altos,
California 94022 (US)

(72) Inventors:
- **Farmwald, Michael**
  Portola Valley,
  California 94028 (US)
- **Horowitz, Mark**
  Palo Alto,
  California 94306 (US)

(74) Representative: **Eisenführ, Speiser & Partner**
Patentanwälte Rechtsanwälte
Postfach 10 60 78
28060 Bremen (DE)

(56) References cited:
US-A- 4 205 373
US-A- 4 630 193
US-A- 4 710 904

EP 1 022 641 B1

Printed by Jouve, 75001 PARIS (FR)

Description

FIELD OF THE INVENTION

[0001] An integrated circuit bus interface for computer and video systems is described which allows high speed transfer of blocks of data, particularly to and from memory devices, with reduced power consumption and increased system reliability. A new method of physically implementing the bus architecture is also described.

BACKGROUND OF THE INVENTION

[0002] semiconductor computer memories have traditionally been designed and structured to use one memory device for each bit, or small group of bits, of any individual computer word, where the word size is governed by the choice of computer. Typical word sizes range from 4 to 64 bits. Each memory device typically is connected in parallel to a series of address lines and connected to one of a series of data lines. When the computer seeks to read from or write to a specific memory location, an address is put on the address lines and some or all of the memory devices are activated using a separate device select line for each needed device. One or more devices may be connected to each data line but typically only a small number of data lines are connected to a single memory device. Thus data line 0 is connected to device(s) 0, data line 1 is connected to device(s) 1, and so on. Data is thus accessed or provided in parallel for each memory read or write operation. For the system to operate properly, every single memory bit in every memory device must operate dependably and correctly.

[0003] To understand the concept of the present invention, it is helpful to review the architecture of conventional memory devices. Internal to nearly all types of memory devices (including the most widely used Dynamic Random Access Memory (DRAM), Static RAM (SRAM) and Read Only Memory (ROM) devices), a large number of bits are accessed in parallel each time the system carries out a memory access cycle. However, only a small percentage of accessed bits which are available internally each time the memory device is cycled ever make it across the device boundary to the external world.

[0004] Referring to Fig. 1, all modern DRAM, SRAM and ROM designs have internal architectures with row (word) lines 5 and column (bit) lines 6 to allow the memory cells to tile a two dimensional area 1. One bit of data is stored at the intersection of each word and bit line. When a particular word line is enabled, all of the corresponding data bits are transferred onto the bit lines. Some prior art DRAMS take advantage of this organization to reduce the number of pins needed to transmit the address. The address of a given memory cell is split into two addresses, row and column, each of which can be multiplexed over a bus only half as wide as the memory cell address of the prior art would have required.

COMPARISON WITH PRIOR ART

[0005] Prior art memory systems have attempted to solve the problem of high speed access to memory with limited success. U.S. Patent No. 3,821,715 (Hoff et. al.), was issued to Intel Corporation for the earliest 4-bit micro-processor. That patent describes a bus connecting a single central processing unit (CPU) with multiple RAMs and ROMs. That bus multiplexes addresses and data over a 4-bit wide bus and uses point-to-point control signals to select particular RAMs or ROMs. The access time is fixed and only a single processing element is permitted. There is no block-mode type of operation, and most important, not all of the interface signals between the devices are bused (the ROM and RAM control lines and the RAM select lines are point-to-point).

[0006] In U.S. Patent No. 4,315,308 (Jackson), a bus connecting a single CPU to a bus interface unit is described. The invention uses multiplexed address, data, and control information over a single 16-bit wide bus. Block-mode operations are defined, with the length of the block sent as part of the control sequence. In addition, variable access-time operations using a "stretch" cycle signal are provided. There are no multiple processing elements and no capability for multiple outstanding requests, and again, not all of the interface signals are bused.

[0007] In U.S. Patent No. 4,449,207 (Kung, et. al.), a DRAM is described which multiplexes address and data on an internal bus. The external interface to this DRAM is conventional, with separate control, address and data connections.

[0008] In U.S. Patent Nos. 4,764,846 and 4,706,166 (Go), a 3-D package arrangement of stacked die with connections along a single edge is described. Such packages are difficult to use because of the point-to-point wiring required to interconnect conventional memory devices with processing elements- Both patents describe complex schemes for solving these problems. No attempt is made to solve the problem by changing the interface.

[0009] In U.S. Patent No. 3,969,706 (Proebsting, et. al.), the current state-of-the-art DRAM interface is described. The address is two-way multiplexed, and there are separate pins for data and control (RAS, CAS, WE, CS). The number of pins grows with the size of the DRAM, and many of the connections must be made point-to-point in a memory system using such DRAMs.

[0010] There are many backplane buses described in the prior art, but not in the combination described or having the

features of this invention. Many backplane buses multiplex addresses and data on a single bus (e.g., the NU bus). ELXSI and others have implemented split-transaction buses (U.S. Patent No. 4,595,923 and 4,481,625 (Roberts)). ELXSI has also implemented a relatively low-voltage-swing current-mode ECL driver (approximately 1 V awing). Address-space registers are implemented on most backplane buses, as is some form of block mode operation.

[0011] Nearly all modern backplane buses implement some type of arbitration scheme, but the arbitration scheme used in this disclosure differs from each of these. U.S. patent Nos. 4,837,682 (Culler), 4,818,985 (Ikeda), 4,779,089 (Theus) and 4,745,548 (Blahut) describe prior art schemes. All involve either log N extra signals, (Theus, Blahut), where N is the number of potential bus requestors, or additional delay to get control of the bus (Ikeda, Culler). None of the buses described in patents or other literature use only bused connections. All contain some point-to-point connections on the backplane. None of the other aspects of this disclosure such as power reduction by fetching each data block from a single device or compact and low-cost 3-D packaging even apply to backplane buses.

[0012] The clocking scheme used in this disclosure has not been used before and in fact would be difficult to implement in backplane buses due to the signal degradation caused by connector stubs. U.S. Patent No. 4,247,817 (Heller) describes a clocking scheme using two clock lines, but relies on ramp-shaped clock signals in contrast to the normal rise-time signals used in the present disclosure.

[0013] In U.S. Patent No. 4,646,279 (Voss), a video RAM is described which implements a parallel-load, serial-out shift register on the output of a DRAM. This generally allows greatly improved bandwidth (and has been extended to 2, 4 and greater width shift-out paths.) The rest of the interfaces to the DRAM (RAS, CAS, multiplexed address, etc.) remain the same as for conventional DRAMS.

[0014] One object of the present invention is to use a new bus interface built into semiconductor devices to support high-speed access to large blocks of data from a single memory device by an external user of the data, such as a microprocessor, in an efficient and cost-effective manner.

[0015] Another object of this invention is to provide devices, especially DRAMs, suitable for use with the bus architecture of the invention.

[0016] D.K Morgan, "The CVAX CMCTL - A CMOS Memory Controller Chip", Digital Technical Journal, No. 7, August 1988, pages 139 to 143 concerns a description of a memory controller chip which is to be used in connection with a "CVAX bus." The document describes that the CVAX memory controller communicates over a synchronous bus with a CPU and over an asynchronous bus with memory modules connected to the controller. The memory modules that are connected to the controller contain hundreds of memory devices. The DRAMs on the memory modules are not discussed with any specificity.

[0017] The "Intel Application Note AP-132", "Designing Memory Systems with the 8K x 8 IRAM", June 1982, pages 3-41 to 3-45; Intel Preliminary Data Sheet "2186- 8192 x8 Bit Integrated RAM", September 1982, pages 3-281; and one undated sheet without a page number relating to the "2186 family" discloses a new type of RAM which aims at combining the best features of SRAM and DRAM. The device is targeted for use in applications with small memory requirements. The disclosed IRAM devices are asynchronous devices (they do not receive a clock signal) that contain the control circuitry necessary for controlling the refresh operations on the memory. Systems known in the timeframe of the IRAM references normally included such refresh control circuitry in an external memory controller.

SUMMARY OF INVENTION

[0018] The invention relates to a system as claimed in claim 1.

[0019] Referring to Fig. 2, a standard DRAM 13, 14, ROM (or SRAM) 12, microprocessor CPU 11, I/O device, disk controller or other special purpose device such as a high speed switch is modified to use a wholly bus-based interface rather than the prior art combination of point-to-point and bus-based wiring used with conventional versions of these devices. The new bus includes clock signals, power and multiplexed address, data and control signals. In a preferred implementation, 8 bus data lines and an AddressValid bus line carry address, data and control information, for memory addresses up to 40 bits wide. Persons skilled in the art will recognize that 16 bus data lines or other numbers of bus data lines can be used to implement the teaching of this invention. The new bus is used to connect elements such as memory, peripheral, switch and processing units.

[0020] In the system of this invention, DRAMs and other devices receive address and control information over the bus and transmit or receive requested data over the same bus. Each memory device contains only a single bus interface with no other signal pins. Other devices that may be included in the system can connect to the bus and other non-bus lines, such as input/output lines. The bus supports large data block transfers and split transactions to allow a user to achieve high bus utilization. This ability to rapidly read or write a large block of data to one single device at a time is an important advantage of this invention.

[0021] The DRAMs that connect to this bus differ from conventional DRAMs in a number of ways. Registers are provided which may store control information, device identification, device-type and other information appropriate for the chip such as the address range for each independent portion of the device. New bus interface circuits must be added

and the internals of prior art DRAM devices need to be modified so they can provide and accept data to and from the bus at the peak data rate of the bus. This requires changes to the column access circuitry in the DRAM, with only a minimal increase in die size. A circuit is provided to generate a low skew internal device clock for devices on the bus, and other circuits provide for demultiplexing input and multiplexing output signals.

[0022] High bus bandwidth is achieved by running the bus at a very high clock rate (hundreds of MHz). This high clock rate is made possible by the constrained environment of the bus. The bus lines are controlled-impedance, doubly-terminated lines. For a data rate of 500 MHz, the maximum bus propagation time is less than 1 ns (the physical bus length is about 10 cm). In addition, because of the packaging used, the pitch of the pins can be very close to the pitch of the pads. The loading on the bus resulting from the individual devices is very small. In a preferred implementation, this generally allows stub capacitances of 1-2 pF and inductances of 0.5 - 2 nH. Each device 15, 16, 17, shown in Figure 3, only has pins on one side and these pins connect directly to the bus 18. A transceiver device 19 can be included to interface multiple units to a higher order bus through pins 20.

[0023] A primary result of the architecture of this invention is to increase the bandwidth of DRAM access. The invention also reduces manufacturing and production costs, power consumption, and increases packing density and system reliability.

BRIEF DESCRIPTION OF THE DRAWINGS

[0024]

Figure 1 is a diagram which illustrates the basic 2-D organization of memory devices.
Figure 2 is a schematic block diagram which illustrates the parallel connection of all bus lines and the serial Reset line to each device in the system.
Figure 3 is a perspective view of a system which illustrates the 3-D packaging of semiconductor devices on the primary bus.
Figure 4 shows the format of a request packet.
Figure 5 shows the format of a retry response from a slave.
Figure 6 shows the bus cycles after a request packet collision occurs on the bus and how arbitration is handled.
Figure 7 shows the timing whereby signals from two devices can overlap temporarily and drive the bus at the same time.
Figure 8 shows the connection and timing between bus clocks and devices on the bus.
Figure 9 is a perspective view showing how transceivers can be used to connect a number of bus units to a transceiver bus.
Figure 10 is a block and schematic diagram of input/output circuitry used to connect devices to the bus.
Figure 11 is a schematic diagram of a clocked sense-amplifier used as a bus input receiver.
Figure 12 is a block diagram showing how the internal device clock is generated from two bus clock signals using a set of adjustable delay lines.
Figure 13 is a timing diagram showing the relationship of signals in the block diagram of Figure 12.
Figure 14 is timing diagram of a preferred means of implementing the reset procedure.
Figure 15 is a diagram illustrating the general organization of a 4 Mbit DRAM divided into 8 subarrays.

DETAILED DESCRIPTION

[0025] The present invention is designed to provide a system containing DRAM semiconductor devices for use with a high speed, multiplexed bus for communication between processing devices and memory devices. The bus can also be used to connect processing devices and other devices, such as I/O interfaces or disk controllers, with or without memory devices on the bus. The bus consists of a relatively small number of lines connected in parallel to each device on the bus. The bus carries substantially all address, data and control information needed by devices for communication with other devices on the bus. In many systems using the present invention, the bus carries almost every signal between every device in the entire system. There is no need for separate device-select lines since device-select information for each device on the bus is carried over the bus. There is no need for separate address and data lines because address and data information can be sent over the same lines. Using the organization described herein, very large addresses (40 bits in the preferred implementation) and large data blocks (1024 bytes) can be sent over a small number of bus lines (8 plus one control line in the preferred implementation).

[0026] Virtually all of the signals needed by a computer system can be sent over the bus. Persons skilled in the art recognize that certain devices, such as CPUs, may be connected to other signal lines and possibly to independent buses, for example a bus to an independent cache memory, in addition to the bus of this embodiment. Certain devices, for example cross-point switches could be connected to multiple, independent buses of this embodiment. In the preferred

implementation, memory devices are provided that have no connections other than the bus connections described herein and CPUs are provided that use the bus of this embodiment as the principal, if not exclusive, connection to memory and to other devices on the bus.

[0027] All modern DRAM, SRAM and ROM designs have internal architectures with row (word) and column (bit) lines to efficiently tile a 2-D area. Referring to Pig. 1, one bit of data is stored at the intersection of each word line 5 and bit line 6. When a particular word line is enabled, all of the corresponding data bits are transferred onto the bit lines. This data, about 4000 bits at a time in a 4 MBit DRAM, is then loaded into column sense amplifiers 3 and held for use by the I/O circuits.

[0028] In the embodiment presented hare, the data from the sense amplifiers is enabled 32 bits at a time onto an internal device bus running at approximately 125 MHz. This internal device bus moves the data to the periphery of the devices where the data is multiplexed into an 8-bit wide external bus interface, running at approximately 500 MBz.

[0029] The bus architecture of this embodiment connects master or bus controller devices, such as CPUs, Direct Memory Access devices (DMAs) or Floating Point Units (PPUs), and slave devices, such as DRAM, SRaM or ROM memory devices. A slave device responds to control signals; a master sends control signals. Persons skilled in the art realize that some devices may behave as both master and slave at various times, depending on the mode of operation and the state of the system. Por example, a memory device will typically have only slave functions, while a DMA controller, disk controller or CPU may include both slave and master functions. Many other semiconductor devices, including I/O devices, disk controllers, or other special purpose devices such as high speed switches can be modified for use with the bus of this embodiment.

[0030] Each semiconductor device contains a set of internal registers, preferably including a device identification (device ID) register, a device-type descriptor register, control registers and other registers containing other information relevant to that type of device. In a preferred implementation, semiconductor devices connected to the bus contain registers which specify the memory addresses contained within that device and access-time registers which store a set of one or more delay times at which the device can or should be available to send or receive data.

[0031] Most of these registers can be modified and preferably are set as part of an initialization sequence that occurs when the system is powered up or reset. During the initialization sequence each device on the bus is assigned a unique device ID number, which is stored in the device ID register. A bus master can then use these device ID numbers to access and set appropriate registers in other devices, including access-time registers, control registers, and memory registers, to configure the system. Each slave may have one or several access-time registers (four in a preferred embodiment). In a preferred embodiment, one access-time register in each slave is permanently or semi-permanently programmed with a fixed value to facilitate certain control functions. A preferred implementation of an initialization sequence is described below in more detail.

[0032] All information sent between master devices and slave devices is sent over the external bus, which, for example, may be 8 bits wide. This is accomplished by defining a protocol whereby a master device, such as a microprocessor, seizes exclusive control of the external bus (i.e., becomes the bus master) and initiates a bus transaction by sending a request packet (a sequence of bytes comprising address and control information) to one or more slave devices on the bus. An address can consist of 16 to 40 or more bits according to the teachings of this embodiment. Each slave on the bus must decode the request packet to see if that slave needs to respond to the packet. The slave that the packet is directed to must then begin any internal processes needed to carry out the requested bus transaction at the requested time. The requesting master may also need to transact certain internal processes before the bus transaction begins. After a specified access time the slave(s) respond by returning one or more bytes (8 bits) of data or by storing information made available from the bus. More than one access time can be provided to allow different types of responses to occur at different times.

[0033] A request packet and the corresponding bus access are separated by a selected number of bus cycles, allowing the bus to be used in the intervening bus cycles by the same or other masters for additional requests or brief bus accesses. Thus multiple, independent accesses are permitted, allowing maximum utilization of the bus for transfer of short blocks of data. Transfers of long blocks of data use the bus efficiently even without overlap because the overhead due to bus address, control and access times is small compared to the total time to request and transfer the block.

Device Address Mapping

[0034] Another unique aspect of this invention is that each memory device is a complete, independent memory sub-system with all the functionality of a prior art memory board in a conventional backplane-bus computer system. Individual memory devices may contain a single memory section or may be subdivided into more than one discrete memory section. Memory devices preferably include memory address registers for each discrete memory section. A failed memory device (or even a subsection of a device) can be "mapped out" with only the lass of a small fraction of the memory, maintaining essentially full system capability. Mapping out bad devices can be accomplished in two ways, both compatible with this invention.

[0035] The preferred method uses address registers in each memory device (or independent discrete portion thereof) to store information which defines the range of bus addresses to which this memory device will respond. This is similar to prior art schemes used in memory boards in conventional backplane bus systems. The address registers can include a single pointer, usually pointing to a block of known size, a pointer and a fixed or variable block size value or two pointers, one pointing to the beginning and one to the end (or to the "top" and "bottom") of each memory block. By appropriate settings of the address registers, a series of functional memory devices or discrete memory sections can be made to respond to a contiguous range of addresses, giving the system access to a contiguous block of good memory, limited primarily by the number of good devices connected to the bus. A block of memory in a first memory device or memory section can be assigned a certain range of addresses, then a block of memory in a next memory device or memory section can be assigned addresses starting with an address one higher (or lower, depending on the memory structure) than the last address of the previous block.

[0036] Preferred devices for use in this embodiment include device-type register information specifying the type of chip, including how much memory is available in what configuration on that device. A master can perform an appropriate memory test, such as reading and writing each memory cell in one or more selected orders, to test proper functioning of each accessible discrete portion of memory (based in part on information like device ID number and device-type) and write address values (up to 40 bits in the preferred embodiment, 10^{12} bytes), preferably contiguous, into device address-space registers. Non-functional or impaired memory sections can be assigned a special address value which the system can interpret to avoid using that memory.

[0037] The second approach puts the burden of avoiding the bad devices on the system master or masters. CPUs and DMA controllers typically have some sort of translation look-aside buffers (TLBS) which map virtual to physical (bus) addresses. With relatively simple software, the TLBs can be programmed to use only working memory (data structures describing functional memories are easily generated)- For masters which don't contain TLBS (for example, a video display generator), a small, simple RAM can be used to map a contiguous range of addresses onto the addresses of the functional memory devices.

[0038] Either scheme works and permits a system to have a significant percentage of non-functional devices and still continue to operate with the memory which remains. This means that systems built with this embodiment will have much improved reliability over existing systems, including the ability to build systems with almost no field failures.

Bus

[0039] The preferred bus architecture of this embodiment comprises 11 signals: BusData [0:7] ; AddrValid; Clk1 and Clk2; plus an input reference level and power and ground lines connected in parallel to each device. Signals are driven onto the bus during conventional bus cycles. The notation "Signal[i:j]" refers to a specific range of signals or lines, for example, BusData[0:7] means BusData0, BusData1, ..., BusData7. The bus lines for BusData[0:7] signals form a byte-wide, multiplexed data/address/control bus. AddrValid is used to indicate when the bus is holding a valid address request, and instructs a slave to decode the bus data as an address and, if the address is included on that slave, to handle the pending request. The two clocks together provide a synchronized, high speed clock for all the devices on the bus. In addition to the bused signals, there is one other line (ResetIn, ResetOut) connecting each device in series for use during initialization to assign every device in the system a unique device ID number (described below in detail).

[0040] To facilitate the extremely high data rate of this external bus relative to the gate delays of the internal logic, the bus cycles are grouped into pairs of even/odd cycles. Note that all devices connected to a bus should preferably use the same even/odd labeling of bus cycles and preferably should begin operations on even cycles. This is enforced by the clocking scheme.

Protocol and Bus Operation

[0041] The bus uses a relatively simple, synchronous, split-transaction, block-oriented protocol for bus transactions. One of the goals of the system is to keep the intelligence concentrated in the masters, thus keeping the slaves as simple as possible (since there are typically many more slaves than masters). To reduce the complexity of the slaves, a slave should preferably respond to a request in a specified time, sufficient to allow the slave to begin or possibly complete a device-internal phase including any internal actions that must precede the subsequent bus access phase. The time for this bus access phase is known to all devices on the bus - each master being responsible for making sure that the bus will be free when the bus access begins. Thus the slaves never worry about arbitrating for the bus. This approach eliminates arbitration in single master systems, and also makes the slave-bus interface simpler.

[0042] In a preferred implementation to initiate a bus transfer over the bus, a master sends out a request packet, a contiguous series of bytes containing address and control information. It is preferable to use a request packet containing an even number of bytes and also preferable to start each packet on an even bus cycle.

[0043] The device-select function is handled using the bus data lines. AddrValid is driven, which instructs all slaves

to decode the request packet address, determine whether they contain the requested address, and if they do, provide the data back to the master (in the case of a read request) or accept data from the master (in the case of a write request) in a data block transfer. A master can also select a specific device by transmitting a device ID number in a request packet. In a preferred implementation, a special device ID number is chosen to indicate that the packet should be interpreted by all devices on the bus. This allows a master to broadcast a message, for example to set a selected control register of all devices with the same value.

[0044] The data block transfer occurs later at a time specified in the request packet control information, preferably beginning on an even cycle. A device begins a data block transfer almost immediately with a device-internal phase as the device initiates certain functions, such as setting up memory addressing, before the bus access phase begins. The time after which a data block is driven onto the bus lines is selected from values stored in slave access-time registers. The timing of data for reads and writes is preferably the same; the only difference is which device drives the bus. For reads, the slave drives the bus and the master latches the values from the bus. For writes the master drives the bus and the selected slave latches the values from the bus.

[0045] In a preferred implementation shown in Figure 4, a request packet 22 contains 6 bytes of data -- 4.5 address bytes and 1.5 control bytes. Each request packet uses all nine bits of the multiplexed data/address lines (AddrValid 23 + BusData[0:7] 24) for all six bytes of the request packet. Setting 23 AddrValid = 1 in an otherwise unused even cycle indicates the start of an request packet (control information). In a valid request packet, AddrValid 27 must be 0 in the last byte. Asserting this signal in the last byte invalidates the request packet. This is used for the collision detection and arbitration logic (described below). Bytes 25-26 contain the first 35 address bits, Address[0:35]. The last byte contains AddrValid 27 (the invalidation switch) and 28, the remaining address bits, Address [36:39], and BlockSize [0:3] (control information).

[0046] The first byte contains two 4 bit fields containing control information, AccessType [0:3], an op code (operation code) which, for example, specifies the type of access, and Master[0:3], a position reserved for the master sending the packet to include its master ID number. Only master numbers 1 through 15 are allowed - master number 0 is reserved for special system commands. Any packet with Master[0:3] = 0 is an invalid or special packet and is treated accordingly.

[0047] The AccessType field specifies whether the requested operation is a read or write and the type of access, for example, whether it is to the control registers or other parts of the device, such as memory. In a preferred implementation, AccessType[0] is a Read/Write switch: if it is a 1, than the operation calls for a read from the slave (the slave to read the requested memory block and drive the memory contents onto the bus) ; if it is a 0, the operation calls for a write into the slave (the slave to read data from the bus and write it to memory). AccessType[1:3] provides up to 8 different access types for a slave. AccessType[1:2] preferably indicates the timing of the response, which is stored in an access-time register, AccessRegN. The choice of access-time register can be selected directly by having a certain op code select that register, or indirectly by having a slave respond to selected op codes with pre-selected access times (see table below). The remaining bit, AccessType[3] may be used to send additional information about the request to the slaves.

[0048] One special type of access is control register access, which involves addressing a selected register in a selected slave. In the preferred implementation, AccessType[1:3] equal to zero indicates a control register request and the address field of the packet indicates the desired control register. For example, the most significant two bytes can be the device ID number (specifying which slave is being addressed) and the least significant three bytes can specify a register address and may also represent or include data to be loaded into that control register. Control register accesses are used to initialize the access-time registers, so it is preferable to use a fixed response time which can be preprogrammed or even hard wired, for example the value in AccessReg0, preferably 8 cycles. Control register access can also be used to initialize or modify other registers, including address registers.

[0049] The method of this embodiment provides for access mode control specifically for the DRAMS. One such access mode determines whether the access is page mode or normal RAS access. In normal mode (in conventional DRAMS and in this invention), the DRAM column sense amps or latches have been precharged to a value intermediate between logical 0 and 1. This precharging allows access to a row in the RAM to begin as soon as the access request for either inputs (writes) or outputs (reads) is received and allows the column sense amps to sense data quickly. In page mode (both conventional and in this invention), the DRAM holds the data in the column sense amps or latches from the previous read or write operation, If a subsequent request to access data is directed to the same row, the DRAM does not need to wait for the data to be sensed (it has been sensed already) and access time for this data is much shorter than the normal access time. Page mode generally allows much faster access to data but to a smaller block of data (equal to the number of sense amps). However, if the requested data is not in the selected row, the access time is longer than the normal access time, since the request must wait for the RAM to precharge before the normal mode access can start. Two access-time registers in each DRAM preferably contain the access times to be used for normal and for page-mode accesses, respectively.

[0050] The access mode also determines whether the DRAM should precharge the sense amplifiers or should save the contents of the sense amps for a subsequent page mode access. Typical settings are "precharge after normal access" and "save after page mode access" but " precharge after page mode access" or "save after normal access" are

allowed, selectable modes of operation. The DRAM can also be set to precharge the sense amps if they are not accessed for a selected period of time.

[0051] In page mode, the data stored in the DRAM sense amplifiers may be accessed within much less time than it takes to read out data in normal mode (~10-20 nS vs. 40-100 nS). This data may be kept available for long periods. However, if these sense amps (and hence bit lines) are not precharged after an access, a subsequent access to a different memory word (row) will suffer a precharge time penalty of about 40-100 nS because the sense amps must precharge before latching in a new value.

[0052] The contents of the sense amps thus may be held and used as a cache, allowing faster, repetitive access to small blocks of data. DRAM-based page-mode caches have been attempted in the prior art using conventional DRAM organizations but they are not very effective because several chips are required per computer word. Such a conventional page-mode cache contains many bits (for example, 32 chips x 4Kbits) but has very few independent storage entries. In other words, at any given point in time the sense amps hold only a few different blocks or memory "locales" (a single block of 4K words, in the example above). Simulations have shown that upwards of 100 blocks are required to achieve high hit rates (>90% of requests find the requested data already in cache memory) regardless of the size of each block. *See, for example,* Anant Agarwal, et. al., "An Analytic Cache Model," *ACM Transactions on Computer Systems,* Vol. 7 (2), pp. 184-215 (May 1989).

[0053] The organization of memory in the present embodiment allows each DRAM to hold one or more (4 for 4MBit DRAMS) separately- addressed and independent blocks of data. A personal computer or workstation with 100 such DRAMs (i.e. 400 blocks or locales) can achieve extremely high, very repeatable hit rates (98-99% on average) as compared to the lower (50-80%), widely varying hit rates using DRAMS organized in the conventional fashion. Further, because of the time penalty associated with the deferred precharge on a "miss" of the page-mode cache, the conventional DRAM-based page-mode cache generally has been found to work less well than no cache at all.

[0054] For DRAM slave access, the access types are preferably used in the following way:

AccessType[1:3]	Use	AccessTime
0	Control Register Access	Fixed, 8[AccessReg0]
1	Unused	Fixed, 8[AccessReg0]
2-3	Unused	AccessReg1
4-5	Page Mode DRAM access	AccessReg2
6-7	Normal DRAM access	AccessReg3

Persons skilled in the art will recognize that a series of available bits could be designated as switches for controlling these access modes. For example:

AccessType[2] = page mode/normal switch

AccessType[3] = precharge/save-data switch

Blocksize[0:3] specifies the size of the data block transfer. If BlockSize[0] is 0, the remaining bits are the binary representation of the block size (0-7). If BlockSize[0] is 1, then the remaining bits give the block size as a binary power of 2, from 8 to 1024. A zero-length block can be interpreted as a special command, for example, to refresh a DRAM without returning any data, or to change the DRAM from page mode to normal access mode or vice-versa.

BlockSize [0:2]	Number of Bytes in Block
0-7	0-7 respectively
8	8
9	16
10	32
11	64
12	128
13	256
14	512
15	1024

Persons skilled in the art will recognize that other block size encoding schemes or values can be used.

[0055] In most cases, a slave will respond at the selected access time by reading or writing data from or to the bus over bus lines BusData[0:7] and AddrValid will be at logical 0. In a preferred embodiment, substantially each memory access will involve only a single memory device, that is, a single block will be read from or written to a single memory device.

Retry Format

[0056] In some cases, a slave may not be able to respond correctly to a request, e.g., for a read or write. In such a situation, the slave should return an error message, sometimes called a N(o)ACK(nowledge) or retry message. The retry message can include information about the condition requiring a retry, but this increases system requirements for circuitry in both slave and masters. A simple message indicating only that an error has occurred allows for a less complex slave, and the master can take whatever action is needed to understand and correct the cause of the error.

[0057] For example, under certain conditions a slave might not be able to supply the requested data. During a page-mode access, the DRAM selected must be in page mode and the requested address must match the address of the data held in the sense amps or latches. Each DRAM can check for this match during a page-mode access. If no match is found, the DRAM begins precharging and returns a retry message to the master during the first cycle of the data block (the rest of the returned block is ignored). The master then must wait for the precharge time (which is set to accommodate the type of slave in question, stored in a special register, PreChargeReg), and then resend the request as a normal DRAM access (AccessType = 6 or 7).

[0058] In the preferred form of the present embodiment, a slave signals a retry by driving AddrValid true at the time the slave was supposed to begin reading or writing data. A master which expected to write to that slave must monitor AddrValid during the write and take corrective action if it detects a retry message. Figure 5 illustrates the format of a retry message 28 which is useful for read requests, consisting of 23 AddrValid=1 with Master[0:3] = 0 in the first (even) cycle. Note that AddrValid is normally 0 for data block transfers and that there is no master 0 (only 1 through 15 are allowed). All DRAMS and masters can easily recognize such a packet as an invalid request packet, and therefore a retry message. In this type of bus transaction all of the fields except for Master[0:3] and AddrValid 23 may be used as information fields, although in the implementation described, the contents are undefined. Persons skilled in the art recognize that another method of signifying a retry message is to add a DataInvalid line and signal to the bus. This signal could be asserted in the case of a NACK.

Bus Arbitration

[0059] In the case of a single master, there are by definition no arbitration problems. The master sends request packets and keeps track of periods when the bus will be busy in response to that packet. The master can schedule multiple requests so that the corresponding data block transfers do not overlap.

[0060] The bus architecture of this embodiment is also useful in configurations with multiple masters. When two or more masters are on the same bus, each master must keep track of all the pending transactions, so each master knows when it can send a request packet and access the corresponding data block transfer. Situations will arise, however, where two or more masters send a request packet at about the same time and the multiple requests must be detected, then sorted out by some sort of bus arbitration.

[0061] There are many ways for each master to keep track of when the bus is and will be busy. A simple method is for each master to maintain a bus-busy data structure, for example by maintaining two pointers, one to indicate the earliest point in the future when the bus will be busy and the other to indicate the earliest point in the future when the bus will be free, that is, the end of the latest pending data block transfer. Using this information, each master can determine whether and when there is enough time to send a request packet (as described above under Protocol) before the bus becomes busy with another data block transfer and whether the corresponding data block transfer will interfere with pending bus transactions. Thus each master must read every request packet and update its bus-busy data structure to maintain information about when the bus is and will be free.

[0062] With two or more masters on the bus, masters will occasionally transmit independent request packets during the same bus cycle. Those multiple requests will collide as each such master drives the bus simultaneously with different information, resulting in scrambled request information and neither desired data block transfer. In a preferred embodiment, each device on the bus seeking to write a logical 1 on a BusData or AddrValid line drives that line with a current sufficient to sustain a voltage greater than or equal to the high-logic value for the system. Devices do not drive lines that should have a logical 0; those lines are simply held at a voltage corresponding to a low-logic value. Each master tests the voltage on at least some, preferably all, bus data and the AddrValid lines so the master can detect a logical '1' where the expected level is '0' on a line that it does not drive during a given bus cycle but another master does drive.

[0063] Another way to detect collisions is to select one or more bus lines for collision signalling. Each master sending a request drives that line or lines and monitors the selected lines for more than the normal drive current (or a logical value of ">1"), indicating requests by more than one master. Persons skilled in the art will recognize that this can be implemented with a protocol involving BusData and AddrValid lines or could be implemented using an additional bus line.

[0064] In the preferred embodiment, each master detects collisions by monitoring lines which it does not drive to see if another master is driving those lines. Referring to Pig. 4, the first byte of the request packet includes the number of each master attempting to use the bus (Master [0:3]). If two masters send packet requests starting at the same point in

time, the master numbers will be logical "or" ed together by at least those masters, and thus one or both of the masters, by monitoring the data on the bus and comparing what it sent, can detect a collision. For instance if requests by masters number 2 (0010) and 5 (0101) collide, the bus will be driven with the value Master[0:3]=7 (0010 + 0101 - 0111). Master number 5 will detect that the signal Master[2] = 1 and master 2 will detect that Master[1] and Master[3] ; 1, telling both masters that a collision has occurred. Another example is masters 2 and 11, for which the bus will be driven with the value Master[0:3]=11 (0010 + 1011 = 1011), and although master 11 can't readily detect this collision, master 2 can. When any collision is detected, each master detecting a collision drives the value of AddrValid 27 in byte 5 of the request packet 22 to 1, which is detected by all masters, including master 11 in the second example above, and forces a bus arbitration cycle, described below.

[0065] Another collision condition may arise where master A sends a request packet in cycle 0 and master B tries to send a request packet starting in cycle 2 of the first request packet, thereby overlapping the first request packet. This will occur from time to time because the bus operates at high speeds, thus the logic in a second-initiating master may not be fast enough to detect a request initiated by a first master in cycle 0 and to react fast enough by delaying its own request. Master B eventually notices that it wasn't supposed to try to send a request packet (and consequently almost surely destroyed the address that master A was trying to send), and, as in the example above of a simultaneous collision, drives a 1 on AddrValid during byte 5 of the first request packet 27 forcing an arbitration. The logic in the preferred implementation is fast enough that a master should detect a request packet by another master by cycle 3 of the first request packet, so no master is likely to attempt to send a potentially colliding request packet later than cycle 2.

[0066] Slave devices not need to detect a collision directly, but they must wait to do anything irrecoverable until the last byte (byte 5) is read to ensure that the packet is valid. A request packet with Master[0:3] equal to 0 (a retry signal) is ignored and does not cause a collision. The subsequent bytes of such a packet are ignored.

[0067] To begin arbitration after a collision, the masters wait a preselected number of cycles after the aborted request packet (4 cycles in a preferred implementation), then use the next free cycle to arbitrate for the bus (the next available even cycle in the preferred implementation). Each colliding master signals to all other colliding masters that it seeks to send a request packet, a priority is assigned to each of the colliding masters, then each master is allowed to make its request in the order of that priority.

[0068] Figure 6 illustrates one preferred way of implementing this arbitration. Each colliding master signals its intent to send a request packet by driving a single BusData line during a single bus cycle corresponding to its assigned master number (1-15 in the present example). During two-byte arbitration cycle 29, byte 0 is allocated to requests 1-7 from masters 1-7, respectively, (bit 0 is not used) and byte 1 is allocated to requests 8-15 from masters 8-15, respectively. At least one device and preferably each colliding master reads the values on the bun during the arbitration cycles to determine and store which masters desire to use the bus. Persons skilled in the art will recognize that a single byte can be allocated for arbitration requests if the system includes more bus lines than masters. More than 15 masters can be accommodated by using additional bus cycles.

[0069] A fixed priority scheme (preferably using the master numbers, selecting lowest numbers first) is then used to prioritize, then sequence the requests in a bus arbitration queue which is maintained by at least one device. These requests are queued by each master in the bus-busy data structure and no further requests are allowed until the bus arbitration queue is cleared. Persons skilled in the art will recognize that other priority schemes can be used, including assigning priority according to the physical location of each master.

System Configuration/Reset

[0070] In the bus-based system of this embodiment, a mechanism is provided to give each device on the bus a unique device identifier (device ID) after power-up or under other conditions as desired or needed by the system. A master can then use this device ID to access a specific device, particularly to set or modify registers of the specified device, including the control and address registers. In the preferred embodiment, one master is assigned to carry out the entire system configuration process. The master provides a series of unique device ID numbers for each unique device connected to the bus system. In the preferred embodiment, each device connected to the bus contains a special device-type register which specifies the type of device, for instance CPU, 4 MBit memory, 64 MBit memory or disk controller. The configuration master should check each device, determine the device type and set appropriate control registers, including access-time registers. The configuration master should check each memory device and set all appropriate memory address registers.

[0071] One means to set up unique device ID numbers is to have each device to select a device ID in sequence and store the value in an internal device ID register. For example, a master can pass sequential device ID numbers through shift registers in each of a series of devices, or pass a token from device to device whereby the device with the token reads in device ID information from another line or lines. In a preferred embodiment, device ID numbers are assigned to devices according to their physical relationship, for instance, their order along the bus.

[0072] In a preferred embodiment, the device ID setting is accomplished using a pair of pins on each device, ResetIn

and ResetOut. These pins handle normal logic signals and are used only during device ID configuration. On each rising edge of the clock, each device copies ResetIn (an input) into a four-stage reset shift register. The output of the reset shift register is connected to ResetOut, which in turn connects to ResetIn for the next sequentially connected device. Substantially all devices on the bus are thereby daisy-chained together. A first reset signal, for example, while ResetIn at a device is a logical 1, or when a selected bit of the reset shift register goes from zero to non-zero, causes the device to hard reset, for example by clearing all internal registers and resetting all state machines. A second reset signal, for example, the falling edge of ResetIn combined with changeable values on the external bus, causes that device to latch the contents of the external bus into the internal device ID register (Device[0:7]).

[0073] To reset all devices on a bus, a master sets the ResetIn line of the first device to a "1" for long enough to ensure that all devices on the bus have been reset (4 cycles times the number of devices -- note that the maximum number of devices on the preferred bus configuration is 256 (8 bits), so that 1024 cycles is always enough time to reset all devices.) Then ResetIn is dropped to "0" and the BusData lines are driven with the first followed by successive device ID numbers, changing after every 4 clock pulses. Successive devices set those device ID numbers into the corresponding device ID register as the falling edge of ResetIn propagates through the shift registers of the daisy-chained devices. Figure 14 shows ResetIn at a first device going low while a master drives a first device ID onto the bus data lines BusData[0:3]. The first device then latches in that first device ID. After four clock cycles, the master changes BusData[0:3] to the next device ID number and ResetOut at the first device goes low, which pulls ResetIn for the next daisy-chained device low, allowing the next device to latch in the next device ID number from BusData [0:3]. In the preferred embodiment, one master is assigned device ID 0 and it is the responsibility of that master to control the ResetIn line and to drive successive device ID numbers onto the bus at the appropriate times. In the preferred embodiment, each device waits two clock cycles after ResetIn goes low before latching in a device ID number from BusData [0:3].

[0074] Persons skilled in the art recognize that longer device ID numbers could be distributed to devices by having each device read in multiple bytes from the bus and latch the values into the device ID register. Persons skilled in the art also recognize that there are alternative ways of getting device ID numbers to unique devices. For instance, a series of sequential numbers could be clocked along the ResetIn line and at a certain time each device could be instructed to latch the current reset shift register value into the device ID register.

[0075] The configuration master should choose and set an access time in each access-time register in each slave to a period sufficiently long to allow the slave to perform an actual, desired memory access. For example, for a normal DRAM access, this time must be longer than the row address strobe (RAS) access time. If this condition is not met, the slave may not deliver the correct data. The value stored in a slave access-time register is preferably one-half the number of bus cycles for which the slave device should wait before using the bus in response to a request. Thus an access time value of '1' would indicate that the slave should not access the bus until at least two cycles after the last byte of the request packet has been received. The value of AccessReg0 is preferably fixed at 8 (cycles) to facilitate access to control registers.

[0076] The bus architecture of this embodiment can include more than one master device. The reset or initialization sequence should also include a determination of whether there are multiple masters on the bus, and if so to assign unique master ID numbers to each. Persons skilled in the art will recognize that there are many ways of doing this. For instance, the master could poll each device to determine what type of device it is, for example, by reading a special register then, for each master device, write the next available master ID number into a special register.

ECC

[0077] Error detection and correction ("ECC") methods well known in the art can be implemented in this system. ECC information typically is calculated for a block of data at the time that block of data is first written into memory. The data block usually has an integral binary size, e.g. 256 bits, and the ECC information uses significantly fewer bits. A potential problem arises in that each binary data block in prior art schemes typically is stored with the ECC bits appended, resulting in a block size that is not an integral binary power.

[0078] In a preferred embodiment, ECC information is stored separately from the corresponding data, which can then be stored in blocks having integral binary size. ECC information and corresponding data can be stored, for example, in separate DRAM devices. Data can be read without ECC using a single request packet, but to write or read error-corrected data requires two request packets, one for the data and a second for the corresponding ECC information. ECC information may not always be stored permanently and in some situations the ECC information may be available without sending a request packet or without a bus data block transfer.

[0079] In a preferred embodiment, a standard data block size can be selected for use with ECC, and the ECC method will determine the required number of bits of information in a corresponding ECC black. RAMs containing ECC information can be programmed to store an access time that is equal to; (1) the access time of the normal RAM (containing data) plus the time to access a standard data block (for corrected data) minus the time to send a request packet (6 bytes); or (2) the access time of a normal RAM minus the time to access a standard ECC block minus the time to send a request

packet. To read a data block and the corresponding ECC block, the master simply issues a request for the data immediately followed by a request for the ECC block . The ECC RAM will wait for the selected access time then drive its data onto the bus right after (in case (1) above)) the data RAM has finished driving out the data block. Persons skilled in the art will recognize that the access time described in case (2) above can be used to drive ECC data before the data is driven onto the bus lines and will recognize that writing data can be done by analogy with the method described for a read. Persons skilled in the art will also recognize the adjustments that must be made in the bus-busy structure and the request packet arbitration methods of this embodiment in order to accommodate these paired ECC requests.

[0080] Since this system is quite flexible, the system designer can choose the size of the data blocks and the number of ECC bits using the memory devices of this embodiment. Note that the data stream on the bus can be interpreted in various ways-For instance the sequence can be 2^n data bytes followed by 2^n ECC bytes (or vice versa), or the sequence can be 2^k iterations of 8 data bytes plus 1 ECC byte. Other information, such as information used by a directory-based cache coherence scheme, can also be managed this way. See, for example, Anant Agarwal, et al., "Scaleable Directory Schemes for Cache Consistency," 15th *International Symposium on Computer Architecture*, June 1988, pp. 280-289. Those skilled in the art will recognize alternative methods of implementing ECC schemes.

Low Power 3-D Packaging

[0081] Another major advantage of a preferred embodiment of this invention is that it drastically reduces the memory system power consumption. Nearly all the power consumed by a prior art DRAM is dissipated in performing row access. By using a single row access in a single RAM to supply all the bits for a block request (compared to a row-access in each of multiple RAMs in conventional memory systems) the power per bit can be made very small. Since the power dissipated by memory devices according to this embodiment is significantly reduced, the devices potentially can be placed much closer together than with conventional designs.

[0082] The bus architecture of this embodiment makes possible an innovative 3-D packaging technology. By using a narrow, multiplexed (time-shared) bus, the pin count for an arbitrarily large memory device can be kept quite small - on the order of 20 pins. Moreover, this pin count can be kept constant from one generation of DRAM density to the next. The low power dissipation allows each package to be smaller, with narrower pin pitches (spacing between the IC pins). With current surface mount technology supporting pin pitches as low as 20 mils, all off-device connections can be implemented on a single edge of the memory device. Semiconductor die useful in this embodiment preferably have connections or pads along one edge of the die which can then be wired or otherwise connected to the package pins with wires having similar lengths. This geometry also allows for very short leads, preferably with an effective lead length of less than 4 mm. Furthermore, this embodiment uses only bused interconnections, i.e., each pad on each device is connected by the bus to the corresponding pad of each other device.

[0083] The use of a low pin count and an edge-connected bus permits a simple 3-D package, whereby the devices are stacked and the bus is connected along a single edge of the stack. The fact that all of the signals are bused is important for the implementation of a simple 3-D structure. Without this, the complexity of the "backplane" would be too difficult to make cost effectively with current technology. The individual devices in a stack of the present embodiment can be packed quite tightly because of the low power dissipated by the entire memory system, permitting the devices to be stacked bumper-to-bumper or top to bottom. Conventional plastic-injection molded small outline (SO) packages can be used with a pitch of about 2.5 mm (100 mils), but the ultimate limit would be the device die thickness, which is about an order of magnitude smaller, 0.2-0.5 mm using current wafer technology.

Bus Electrical Description

[0084] By using devices with very low power dissipation and close physical packing, the bus can be made quite short, which in turn allows for short propagation times and high data rates. The bus of a preferred embodiment consists of a set of resistor-terminated controlled impedance transmission lines which can operate up to a data rate of 500 MHz (2 ns cycles). The characteristics of the transmission lines are strongly affected by the loading caused by the DRAMs (or other slaves) mounted on the bus. These devices add lumped capacitance to the lines which both lowers the impedance of the lines and decreases the transmission speed. In the loaded environment, the bus impedance is likely to be on the order of 25 ohms and the propagation velocity about c/4 (c = the speed of light) or 7.5 cm/ns. To operate at a 2 ns data rate, the transit time on the bus should preferably be kept under 1 ns, to leave 1 ns for the setup and hold time of the input receivers (described below) plus clock skew. Thus the bus lines must be kept quite short, under about 8 cm for maximum performance. Lower performance systems may have much longer lines, e.g. a 4 ns bus may have 24 cm lines (3 ns transit time, 1 ns setup and hold time).

[0085] In the preferred embodiment, the bus uses current source drivers. Each output must be able to sink 50 mA, which provides an output swing of about 500 mV or more. In the preferred embodiment, the bus is active low. The unasserted state (the high value) is preferably considered a logical zero, and the asserted value (low state) is therefore

a logical 1. Those skilled in the art understand that the method of this embodiment can also be implemented using the opposite logical relation to voltage. The value of the unasserted state is set by the voltage on the termination resistors, and should be high enough to allow the outputs to act as current sources, while being as low as possible to reduce power dissipation. These constraints may yield a termination voltage about 2V above ground in the preferred implementation. Current source drivers cause the output voltage to be proportional to the sum of the sources driving the bus.

[0086] Referring to Fig.7, although there is no stable condition where two devices drive the bus at the same time, conditions can arise because of propagation delay on the wires where one device, A 41, can start driving its part of the bus 44 while the bus is still being driven by another device, B 42 (already asserting a logical 1 on the bus). In a system using current drivers, when B 42 is driving the bus (before time 46), the value at points 44 and 45 is logical 1. If B 42 switches off at time 46 just when A 41 switches on, the additional drive by device A 41 causes the voltage at the output 44 of A 41 to drop briefly below the normal value. The voltage returns to its normal value at time 47 when the effect of device B 42 turning off is felt. The voltage at point 45 goes to logical 0 when device B 42 turns off, then drops at time 47 when the effect of device A 41 turning on is felt. Since the logical 1 driven by current from device A 41 is propagated irrespective of the previous value on the bus, the value on the bus is guaranteed to settle after one time of flight (t_f) delay, that is, the time it takes a signal to propagate from one end of the bus to the other. If a voltage drive was used (as in ECL wired-ORing), a logical 1 on the bus (from device B 42 being previously driven) would prevent the transition put out by device A 41 being felt at the most remote part of the system, e.g., device 43, until the turnoff waveform from device B 42 reached device A 41 plus one time of flight delay, giving a worst case settling time of twice the time of flight delay.

Clocking

[0087] Clocking a high speed bus accurately without introducing error due to propagation delays can be implemented by having each device monitor two bus clock signals and then derive internally a device clock, the true system clock. The bus clock information can be sent on one or two lines to provide a mechanism for each bused device to generate an internal device clock with zero skew relative to all the other device clocks. Referring to Figure 8, in the preferred implementation, a bus clock generator 50 at one end of the bus propagates an early bus clock signal in one direction along the bus, for example on line 53 from left to right, to the far end of the bus. The same clock signal then is passed through the direct connection shown to a second Line 54, and returns as a late bus clock signal along the bus from the far end to the origin, propagating from right to left. A single bus clock line can be used if it is left unterminated at the far end of the bus, allowing the early bus clock signal to reflect back along the same line as a late bus clock signal.

[0088] Figure 8b illustrates how each device 51, 52 receives each of the two bus clock signals at a different time (because of propagation delay along the wires), with constant midpoint in time between the two bus clocks along the bus. At each device 51, 52, the rising edge 55 of Clock1 53 is followed by the rising edge 56 of Clock2 54. Similarly, the falling edge 57 of Clock1 53 is followed by the falling edge 58 of Clock2 54. This waveform relationship is observed at all other devices along the bus. Devices which are closer to the clock generator have a greater separation between Clock1 and Clock2 relative to devices farther from the generator because of the longer time required for each clock pulse to traverse the bus and return along line 54, but the midpoint in time 59, 60 between corresponding rising or falling edges is fixed because, for any given device, the length of each clock line between the far end of the bus and that device is equal. Each device must sample the two bus clocks and generate its own internal device clock at the midpoint of the two.

[0089] Clock distribution problems can be further reduced by using a bus clock and device clock rate equal to the bus cycle data rate divided by two, that is, the bus clock period is twice the bus cycle period. Thus a 500 MHz bus preferably uses a 250 MHz clock rate. This reduction in frequency provides two : benefits. First it makes all signals on the bus have the same worst case data rates -- data on a 500 MHz bus can only change every 2 ns. Second, clocking at half the bus cycle data rate makes the labeling of the odd and even bus cycles trivial, for example, by defining even cycles to be those when the internal device clock is 0 and odd cycles when the internal device clock is 1.

Multiple Buses

[0090] The limitation on bus length described above restricts the total number of devices that can be placed on a single bus. Using 2.5 mm spacing between devices, a single 8 cm bus will hold about 32 devices. Persons skilled in the art will recognize certain applications of the present embodiment wherein the overall data rate on the bus is adequate but memory or processing requirements necessitate a much larger number of devices (many more than 32). Larger systems can easily be built using the teachings of this embodiment by using one or more memory subsystems, designated primary bus units, each of which consists of two or more devices, typically 32 or close to the maximum allowed by bus design requirements, connected to a transceiver device.

[0091] Referring to Figure 9, each primary bus unit can be mounted on a single circuit board 66, sometimes called a memory stick. Each transceiver device 19 in turn connects to a transceiver bus 65, similar or identical in electrical and

other respects to the primary bus 18 described at length above. In a preferred implementation, all masters are situated on the transceiver bus so there are no transceiver delays between masters and all memory devices are on primary bus units so that all memory accesses experience an equivalent transceiver delay, but persons skilled in the art will recognize how to implement systems which have masters on more than one bus unit and memory devices on the transceiver bus as well as on primary bus units. In general, each teaching of this embodiment which refers to a memory device can be practiced using a transceiver device and one or more memory devices on an attached primay bus unit. Other devices, generically referred to as peripheral devices, including disk controllers, video controllers or I/O devices can also be attached to either the transceiver bus or a primary bus unit, as desired. Persons skilled in the art will recognize how to use a single primary bus unit or multiple primary bus units as needed with a transceiver bus in certain system designs.

[0092] The transceivers are quite simple in function. They detect request packets on the transceiver bus and transmit them to their primary bus unit. If the request packet calls for a write to a device on a transceiver's primary bus unit, that transceiver keeps track of the access time and block size and forwards all data from the transceiver bus to the primary bus unit during that time. The transceivers also watch their primary bus unit, forwarding any data that occurs there to the transceiver bus. The high speed of the buses means that the transceivers will need to be pipelined, and will require an additional one or two cycle delay for data to pass through the transceiver in either direction. Access times stored in masters on the transceiver bus must be increased to account for transceiver delay but access times stored in slaves on a primary bus unit should not be modified.

[0093] Persons skilled in the art will recognize that a more sophisticated transceiver can control transmissions to and from primary bus units. An additional control line, TrncvrRw can be bused to all devices on the transceiver bus, using that line in conjunction with the AddrValid line to indicate to all devices on the transceiver bus that the information on the data lines is: 1) a request packet, 2) valid data to a slave, 3) valid data from a slave, or 4) invalid data (or idle bus). Using this extra control line obviates the need for the transceivers to keep track of when data needs to be forwarded from its primary bus to the transceiver bus - all transceivers send all data from their primary bus to the transceiver bus whenever the control signal indicates condition 2) above. In a preferred implementation if AddrValid and TrncvrRW are both low, there is no bus activity and the transceivers should remain in an idle state. A controller sending a request packet will drive AddrValid high, indicating to all devices on the transceiver bus that a request packet is being sent which each transceiver should forward to its primary bus unit. Each controller seeking to write to a slave should drive both AddrValid and TrncvrRW high, indicating valid data for a slave is present on the data lines. Each transceiver device will then transmit all data from the transceiver bus lines to each primary bus unit. Any controller expecting to receive information from a slave should also drive the TrncvrRW line high, but not drive AddrValid, thereby indicating to each transceiver to transmit any data coming from any slave on its primary local bus to the transceiver bus. A still more sophisticated transceiver would recognize signals addressed to or coming from its primary bus unit and transmit signals only at requested times.

[0094] An example of the physical mounting of the transceivers is shown in Figure 9. One important feature of this physical arrangement is to integrate the bus of each transceiver 19 with the original bus of DRAMs or other devices IS, 16, 17 on the primary bus unit 66. The transceivers 19 have pins on two sides, and are preferably mounted flat on the primary bus unit with a first set of pins connected to primary bus 18. A second set of transceiver pins 20, preferably orthogonal to the first set of pins, are oriented to allow the transceiver 19 to be attached to the transceiver bus 65 in much the same way as the DRAMs were attached to the primary bus unit. The transceiver bus can be generally planar and in a different plane, preferably orthogonal to the plane of each primary bus unit. The transceiver bus can also be generally circular with primary bus units mounted perpendicular and tangential to the transceiver bus.

[0095] Using this two level scheme allows one to easily build a system that contains over 500 slaves (16 buses of 32 DRAMs each). Persons skilled in the art can modify the device ID scheme described above to accommodate more than 256 devices, for example by using a longer device ID or by using additional registers to hold some of the device ID. This scheme can be extended in yet a third dimension to make a second-order transceiver bus, connecting multiple transceiver buses by aligning transceiver bus units parallel to and on top of each other and busing corresponding signal lines through a suitable transceiver. Using such a second-order transceiver bus, one could connect many thousands of slave devices into what is effectively a single bus.

Device Interface

[0096] The device interface to the high-speed bus can be divided into three main parts. The first part is the electrical interface. This part includes the input receivers, bus drivers and clock generation circuitry. The second part contains the address comparison circuitry and timing registers. This part takes the input request packet and determines if the request is for this device, and if it is, starts the internal access and delivers the data to the pins at the correct time. The final part, specifically for memory devices such as DRAMs, is the DRAM column access path. This part needs to provide bandwidth into and out of the DRAM sense amps greater than the bandwidth provided by conventional DRAMs. The implementation of the electrical interface and DRAM column access path are described in more detail in the following sections. Persons

skilled in the art recognize how to modify prior-art address comparison circuitry and prior-art register circuitry.

Electrical Interface - Input/Output Circuitry

[0097] A block diagram of the preferred input/output circuit for address/data/control lines is shown in Figure 10. This circuitry is particularly well-suited for use in DRAM devices but it can be used or modified by one skilled in the art for use in other devices connected to the bus of this embodiment. It consists of a set of input receivers 71, 72 and output driver 76 connected to input/output line 69 and pad 75 and circuitry to use the internal clock 73 and internal clock complement 74 to drive the input interface. The clocked input receivers take advantage of the synchronous nature of the bus. To further reduce the performance requirements for device input receivers, each device pin, and thus each bus line, is connected to two clocked receivers, one to sample the even cycle inputs, the other to sample the odd cycle inputs. By thus de-multiplexing the input 70 at the pin, each clocked amplifier is given a full 2 ns cycle to amplify the bus low-voltage-swing signal into a full value CMOS logic signal. Persons skilled in the art will recognize that additional clocked input receivers can be used. For example, four input receivers could be connected to each device pin and clocked by a modified internal device clock to transfer sequential bits from the bus to internal device circuits, allowing still higher external bus speeds or still longer settling times to amplify the bus low-voltage-swing signal into a full value CMOS logic signal ,

[0098] The output drivers are quite simple, and consist of a single NMOS pulldown transistor 76. This transistor is sized so that under worst case conditions it can still sink the 50 mA required by the bus. For 0.8 micron CMOS technology, the transistor will need to be about 200 microns long. Overall bus performance can be improved by using feedback techniques to control output transistor current so that the current through the device is roughly 50 mA under all operating conditions, although this is not absolutely necessary for proper bus operation. An example of one of many methods known to persons skilled in the art for using feedback techniques to control current is described in Hans Schumacher, et al. , "CMOS Subnanosecond True-ECL Output Buffer, *J. Solid State Circuits*, Vol. 25 (1), pp. 150-154 (Feb. 1990). Controlling this current improves performance and reduces power dissipation. This output driver which can be operated at 500 MHz, can in turn be controlled by a suitable multiplexer with two or more (preferably four) inputs connected to other internal chip circuitry, all of which can be designed according to well known prior art.

[0099] The input receivers of every slave must be able to operate during every cycle to determine whether the signal on the bus is a valid request packet. This requirement leads to a number of constraints on the input circuitry. In addition to requiring small acquisition and resolution delays, the circuits must take little or no DC power, little AC power and inject very little current back into the input or reference lines. The standard clocked DRAM sense amp shown in Figure 11 satisfies all these requirements except the need for low input currents. When this sense amp goes from sense to sample, the capacitance of the internal nodes 83 and 84 in Figure11 is discharged through the reference line 68 and input 69, respectively. This particular current is small, but the sum of such currents from all the inputs into the reference lines summed over all devices can be reasonably large.

[0100] The fact that the sign of the current depends upon on the previous received data makes matters worse. One way to solve this problem is to divide the sample period into two phases. During the first phase, the inputs are shorted to a buffered version of the reference level (which may have an offset). During the second phase, the inputs are connected to the true Inputs. This scheme does not remove the input current completely, since the input must still charge nodes 83 and 84 from the reference value to the current input value, but it does reduce the total charge required by about a factor of 10 (requiring only a 0.25v change rather than a 2.5V change). Persons skilled in the art will recognize that many other methods can be used to provide a clocked amplifier that will operate on very low input currents.

[0101] One important part of the input/output circuitry generates an internal device clock based on early and late bus clocks. Controlling clock skew (the difference in clock timing between devices) is important in a system running with 2 ns cycles, thus the internal device clock is generated so the input sampler and the output driver operate as close in time as possible to midway between the two bus clocks.

[0102] A block diagram of the internal device clock generating circuit is shown in Figure 12 and the corresponding timing diagram in Figure 13. The basic idea behind this circuit is relatively simple. A DC amplifier 102 is used to convert the small-swing bus clock into a full-swing CMOS signal. This signal is then fed into a variable delay line 103. The output of delay line 103 feeds three additional delay lines: 104 having a fixed delay; 105 having the same fixed delay plus a second variable delay; and 106 having the same fixed delay plus one half of the second variable delay. The outputs 107, 108 of the delay lines 104 and 105 drive clocked input receivers 101 and 111 connected to early and late bus clock inputs 100 and 110, respectively. These input receivers 101 and 111 have the same design as the receivers described above and shown in Fig. 11. Variable delay lines 103 and 105 are adjusted via feedback lines 116, 115 so that input receivers 101 and 111 sample the bus clocks just as they transition. Delay lines 103 and 105 are adjusted so that the falling edge 120 of output 107 precedes the falling edge 121 of the early bus clock, Clock1 53, by an amount of time 128 equal to the delay in input sampler 101. Delay line 108 is adjusted in the same way so that falling edge 122 precedes the falling edge 123 of late bus clock, Clock2 54, by the delay 128 in input sampler 111.

[0103] Since the outputs 107 and 108 are synchronized with the two bus clocks and the output 73 of the last delay line 106 is midway between outputs 107 and 108, that is, output 73 follows output 107 by the same amount of time 129 that output 73 precedes output 108, output 73 provides an internal device clock midway between the bus clocks. The falling edge 124 of internal device clock 73 precedes the time of actual input sampling 125 by one sampler delay. Note that this circuit organization automatically balances the delay in substantially all device input receivers 71 and 72 (Fig. 10), since outputs 107 and 108 are adjusted so the bus clocks are sampled by input receivers 101 and 111 just as the bus clocks transition.

[0104] In the preferred embodiment, two sets of these delay lines are used, one to generate the true value of the internal device clock 73, and the other to generate the complement 74 without adding any inverter delay. The dual circuit allows generation of truly complementary clocks, with extremely small skew. The complement internal device clock is used to clock the 'even' input receivers to sample at time 127, while the true internal device clock is used to clock the 'odd' input receivers to sample at time 125. The true and complement internal device clocks are also used to select which data is driven to the output drivers. The gate delay between the internal device clock and output circuits driving the bus is slightly greater than the corresponding delay for the input circuits, which means that the new data always will be driven on the bus slightly after the old data has been sampled.

DRAM Column Access Modification

[0105] A block diagram of a conventional 4 MBit DRAM 130 is shown in Figure 15. The DRAM memory array is divided into a number of subarrays 150-157, for example, 8. Each subarray is divided into arrays 148, 149 of memory cells. Row address selection is performed by decoders 146. A column decoder 147A, 147B, including column sense amps on either side of the decoder, runs through the core of each subarray. These column sense amps can be set to precharge or latch the most-recently stored value, as described in detail above. Internal I/O lines connect each set of sense-amps, as gated by corresponding column decoders, to input and output circuitry connected ultimately to the device pins. These internal I/O lines are used to drive the data from the selected bit lines to the data pins (some of pins 131-145), or to take the data from the pins and write the selected bit lines. Such a column access path organized by prior art constraints does not have sufficient bandwidth to interface with a high speed bus. The method of this embodiment does not require changing the overall method used for column access, but does change implementation details. Many of these details have been implemented selectively in certain fast memory devices but never in conjunction with the bus architecture of this embodiment.

[0106] Running the internal I/O lines in the conventional way at high bus cycle rates is not possible. In the preferred method, several (preferably 4) bytes are read or written during each cycle and the column access path is modified to run at a lower rate (the inverse of the number of bytes accessed per cycle, preferably 1/4 of the bus cycle rate). Three different techniques are used to provide the additional internal I/O lines required and to supply data to memory cells at this rate. First, the number of I/O bit lines in each subarray running through the column decoder 147 is increased, for example, to 16, eight for each of the two columns of column sense amps and the column decoder selects one set of columns from the "top" half 148 of subarray 150 and one set of columns from the "bottom" half 149 during each cycle, where the column decoder selects one column sense amp per I/O bit line. Second, each column I/O line is divided into two halves, carrying data independently over separate internal I/O lines from the left half 147A and right half 147B of each subarray (dividing each subarray into quadrants) and the column decoder selects sense amps from each right and left half of the subarray, doubling the number of bits available at each cycle. Thus each column decode selection turns on n column sense amps, where n equals four (top left and right, bottom left and right quadrants) times the number of I/O lines in the bus to each subarray quadrant (8 lines each x 4=32 lines in the preferred implementation). Finally, during each RAS cycle, two different subarrays, e.g. 157 and 153, are accessed. This doubles again the available number of I/O lines containing data. Taken together, these changes increase the internal I/O bandwidth by at least a factor of 8. Four internal buses are used to route these internal I/O lines. Increasing the number of I/O lines and then splitting them in the middle greatly reduces the capacitance of each internal I/O line which in turn reduces the column access time, increasing the column access bandwidth even further.

[0107] The multiple, gated input receivers described above allow high speed input from the device pins onto the internal I/O lines and ultimately into memory. The multiplexed output driver described above is used to keep up with the data flow available using these techniques. Control means are provided to select whether information at the device pins should be treated as an address, and therefore to be decoded, or input or output data to be driven onto or read from the internal I/O lines.

[0108] Each subarray can access 32 bits per cycle, 16 bits from the left subarray and 16 from the right subarray. With a I/O lines per sense-amplifier column and accessing two subarrays at a time, the DRAM can provide 64 bits per cycle. This extra I/O bandwidth is not needed for reads (and is probably not used), but may be needed for writes. Availability of write bandwidth is a more difficult problem than read bandwidth because over-writing a value in a sense-amplifier may be a slow operation, depending on how the sense amplifier is connected to the bit line. The extra set of internal I/O

lines provides some bandwidth margin for write operations.

Claims

1. A system comprising:

   a bus that includes a plurality of bus lines for carrying substantially all address, data and control information needed by each semiconductor device coupled to the bus for communication with substantially every other semiconductor device connected to the bus, wherein the bus uses address multiplexing to convey a single memory address; and
   a plurality of synchronous dynamic random access memory (DRAM) semiconductor devices coupled to the bus, each DRAM of the plurality of synchronous DRAM semiconductor devices including:

      connection means adapted to connect the DRAM to the bus;
      clock receiver circuitry (101, 111) for receiving a clock signal (53, 54);
      a programmable access-time register for storing a value which is representative of a number of clock cycles of the clock signal (53, 54) to transpire after which the DRAM responds to a read request received synchronously with respect to the clock signal, the programmable access-time register being accessible to the bus through the connection means, wherein data is transmitted to the programmable access-time register over the bus to set the value in the programmable access-time register; and
      a plurality of output drivers (76) for outputting data onto the bus (18, 65) in response to the read request, wherein the output drivers (76) output the data on the bus (18, 65) after the number of clock cycles of the clock signal transpire and synchronously with respect to the clock signal (53, 54), so that the read request and the corresponding response are separated by the number of clock cycles as selected by the value stored in the programmable access-time register, wherein each output driver of the plurality of output drivers (76) outputs the data onto the bus (18, 65) at a bus cycle data rate that is twice the rate of the clock signal.

2. The system of claim 1, wherein, for each DRAM, the value is stored in the programmable access-time register after power is applied to the DRAM and in response to a set register operation.

3. The system of any of the preceding claims,
   wherein, in response to a read request, the plurality of output drivers (76) output an amount of data defined by block size information received by each DRAM synchronously with respect to the clock signal (53, 54).

4. The system according to any one of the preceding claims, further **characterised in that** each DRAM comprises:

   an internal device clock generating circuit, coupled to the clock receiver circuitry, to generate an internal clock signal used to output data in response to the read request, wherein the internal device clock generating circuit monitors and controls a timing relationship between the data to be output and the clock signal received by the DRAM.

5. The system of claim 4, wherein the internal device clock generating circuit is to generate a complement internal dock signal, wherein the internal clock signal and the complement internal clock signal are used to select which data are driven to the plurality of output drivers.

6. The system of claim 5,
   wherein the plurality of output drivers (76) output a first portion of the data in response to a rising edge transition of the internal clock signal.

7. The system according to any one of claims 4, 5, and 6, wherein the internal device clock generating circuit uses a feedback line to adjust a variable delay line in generating the internal clock signal.

8. The system according to one of the preceding claims, wherein each DRAM further includes sense amplifiers used for reading the data from the memory array, wherein when precharge information received from a controller over the bus indicates that a precharge operation should be performed, automatically precharging a portion of the memory array and the sense amplifiers as a part of execution of the read request.

Patentansprüche

1. System umfassend:

einen Bus, welcher eine Mehrzahl von Busleitungen zum Übertragen im Wesentlichen aller Adressen, Daten und Steuerinformationen, welche von jeder Halbleitervorrichtung benötigt wird, die zur Kommunikation an den Bus gekoppelt ist, mit im Wesentlichen jeder anderen Halbleitervorrichtung, die an den Bus angeschlossen ist, wobei der Bus Adressmultiplexen verwendet, um eine einzelne Speicheradresse zu übermitteln, und
eine Mehrzahl von synchronen dynamischen Direktzugriffsspeicher (DRAM)-Halbleitervorrichtungen, welche an den Bus gekoppelt sind, wobei jedes DRAM aus der Mehrzahl von synchronen DRAM-Halbleitervorrichtungen beinhaltet:

Kopplungsmittel, welche angepasst sind, um das DRAM an den Bus zu koppeln,
einen Taktempfängerschaltung (101, 111) zum Empfangen eines Taktsignals (53, 54),
ein programmierbares Zugriffszeit-Register zum Speichern eines Wertes, welcher repräsentativ für eine Anzahl von Taktzyklen des Taktsignals (53, 54) ist, die verstreichen sollen, bevor das DRAM auf die Leseanfrage antwortet, die synchron bezüglich des Taktsignals empfangen worden ist, wobei das programmierbare Zugriffszeit-Register für den Bus über die Kopplungsmittel zugänglich ist, wobei Daten zu dem programmierbaren Zugriffszeit-Register über den Bus übertragen werden, um den Wert in dem programmierbaren Zugriffszeit-Register einzustellen, und
eine Mehrzahl von Ausgabe-Treibern (76) zum Ausgeben von Daten auf den Bus (18, 65) in Reaktion auf eine Leseanforderung, wobei die Ausgabe-Treiber (76), nach Verstreichen der Anzahl von Taktzyklen des Taktsignals, und synchron mit dem Taktsignal (53, 54) die Daten auf den Bus (18, 65) ausgeben, so dass die Leseanforderung und die korrespondierende Antwort durch die Anzahl der Taktzyklen, wie sie durch den gespeicherten Wert in dem programmierbaren Zugriffszeit-Register bestimmt ist, voneinander beabstandet sind, wobei jeder Ausgabe-Treiber aus der Mehrzahl von Ausgabe-Treibern (76) die Daten auf den Bus (18, 65) mit einer Buszyklus-Datenrate ausgibt, die das Doppelte der Rate des Taktsignals ist.

2. System nach Anspruch 1, wobei für jedes DRAM der Wert in dem programmierbaren Zugriffszeit-Register gespeichert wird, nachdem die Spannungsversorgung an das DRAM angelegt wird und als Antwort auf eine Operation zum Setzen des Registers.

3. System nach einem der vorstehenden Ansprüche, wobei in Reaktion auf die Leseanfrage die Mehrzahl der Ausgabe-Treiber (76) eine Datenmenge ausgibt, die durch eine Blockgrößeninformation definiert ist, die von jedem DRAM synchron bezüglich des Taktsignals (53, 54) empfangen wird.

4. System nach einem der vorstehenden Ansprüche, weiterhin dadurch bestimmt, dass jedes DRAM umfasst:

eine vorrichtungsinterne Takterzeugungsschaltung, die mit der Taktempfängerschaltung gekoppelt ist, um ein internes Taktsignal zu erzeugen, welches verwendet wird, um Daten als Antwort auf die Leseanforderung auszugeben, wobei die vorrichtungsinterne Takterzeugungsschaltung ein Timingverhältnis zwischen den auszugebenden Daten und dem von dem DRAM empfangenen Taktsignal überwacht und steuert.

5. System nach Anspruch 4, wobei die vorrichtungsinterne Takterzeugungsschaltung ein komplementäres internes Taktsignal erzeugt, wobei das interne Taktsignal und das komplementäre interne Taktsignal verwendet werden, um auszuwählen, welche Daten an die Mehrzahl von Ausgabe-Treibern ausgegeben werden.

6. System nach Anspruch 5, wobei die Mehrzahl der Ausgabe-Treiber (76) einen ersten Teil der Daten in Reaktion auf die steigende Flanke eines Pegelwechsel des internen Taktsignals ausgeben.

7. System nach einem der Ansprüche 4, 5 oder 6, wobei die vorrichtungsinterne Takterzeugungsschaltung eine Rückkopplungsleitung verwendet, um eine variable Verzögerungsleitung beim Erzeugen des internen Taktsignals einzustellen.

8. System nach einem der vorstehenden Ansprüche, wobei jedes DRAM weiterhin Leseverstärker beinhaltet, die zum Lesen der Daten aus der Speichermatrix verwendet werden, wobei, wenn Voraufladeinformationen, welche von einem Controller über den Bus empfangen wird, anzeigt, dass ein Voraufladevorgang ausgeführt werden soll, automatisches Voraufladen eines Abschnitts des Speichermatrix und der Leseverstärker als Teil des Ausführens

der Leseanforderung aufgeführt werden.

Revendications

1. Système comprenant :

   un bus qui inclut une pluralité de lignes omnibus pour transporter sensiblement toutes les informations de commande, de données et d'adresse requises pour chaque dispositif semi-conducteur couplé au bus pour une communication avec sensiblement chaque autre dispositif semi-conducteur connecté au bus, dans lequel le bus utilise le multiplexage d'adresses pour véhiculer une seule adresse de mémoire ; et
   une pluralité de dispositifs semi-conducteurs de mémoire vive dynamique (DRAM) synchronisés couplés au bus, chaque DRAM de la pluralité de dispositifs semi-conducteurs de DRAM synchronisés incluant :

      un moyen de connexion adapté pour connecter la DRAM au bus ;
      un ensemble de circuits (101, 111) de réception d'horloge pour recevoir un signal (53, 54) d'horloge ;
      un registre de temps d'accès programmable pour stocker une valeur qui est représentative d'un nombre de cycles d'horloge du signal (53, 54) d'horloge pour arriver après que la DRAM répond à une demande de lecture reçue de façon synchronisée relativement au signal d'horloge, le registre de temps d'accès programmable étant accessible au bus à travers le moyen de connexion, dans lequel les données sont transmises au registre de temps d'accès programmable sur le bus pour configurer la valeur dans le registre de temps d'accès programmable ; et
      une pluralité de pilotes (76) de sortie pour sortir les données sur le bus (18, 65) en réponse à la demande de lecture, dans laquelle les pilotes (76) de sortie sortent les données sur le bus (18, 65) après que le nombre de cycles d'horloge du signal d'horloge arrive et de façon synchronisée relativement au signal (53, 54) d'horloge, de telle sorte que la demande de lecture et la réponse correspondante sont séparées par le nombre de cycles d'horloge tel que sélectionné par la valeur stockée dans le registre de temps d'accès programmable, dans lequel chaque pilote de sortie de la pluralité de pilotes (76) de sortie sort les données sur le bus (18, 65) à un débit de données de cycle de bus qui est le double du débit du signal d'horloge.

2. Système selon la revendication 1, dans lequel, pour chaque DRAM, la valeur est stockée dans le registre de temps d'accès programmable après que l'alimentation est fournie à la DRAM et en réponse à une opération de registre d'ensemble.

3. Système de l'une quelconque des revendications précédentes, dans lequel, en réponse à une demande de lecture, la pluralité de pilotes (76) de sortie sort une quantité de données définie par des informations de taille de bloc reçues par chaque DRAM de façon synchronisée relativement au signal (53, 54) d'horloge.

4. Système selon l'une quelconque des revendications précédentes, en outre **caractérisé en ce que** chaque DRAM comprend :

   un circuit générateur d'horloge de dispositif interne, couplé à l'ensemble de circuits de réception d'horloge, pour générer un signal d'horloge interne utilisé pour sortir des données en réponse à la demande de lecture, dans lequel le circuit générateur d'horloge de dispositif interne surveille et contrôle une relation de synchronisation entre les données destinées à être sorties et le signal d'horloge reçu par la DRAM.

5. Système selon la revendication 4, dans lequel le circuit générateur d'horloge interne sert à générer un signal d'horloge interne de complément, dans lequel le signal d'horloge interne et le signal d'horloge interne de complément sont utilisés pour sélectionner quelles données sont envoyées à la pluralité de pilotes de sortie.

6. Système selon la revendication 5, dans lequel la pluralité de pilotes (76) de sortie sort une première partie des données en réponse à une transition de bord croissante du signal d'horloge interne.

7. Système selon l'une quelconque des revendications 4, 5 et 6, dans lequel le circuit générateur d'horloge de dispositif interne utilise une ligne de rétroaction pour ajuster une ligne à retard variable en générant le signal d'horloge interne.

8. Système selon l'une quelconque des revendications précédentes, dans lequel chaque DRAM inclut en outre des amplificateurs de directions utilisés pour lire les données de la matrice mémoire, dans laquelle lorsque les informa-

tions de précharge reçues d'un contrôleur sur le bus indiquent qu'une opération de précharge devrait être exécutée, préchargeant de façon automatique une partie de la matrice mémoire et les amplificateurs de directions en tant que partie d'exécution de la demande de lecture.

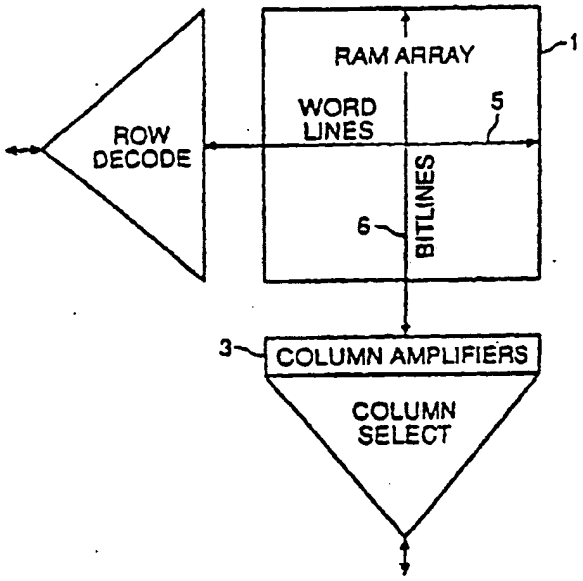
RAM ARRAY
1
WORD LINES
5
ROW DECODE
BITLINES
6
3
COLUMN AMPLIFIERS
COLUMN SELECT

FIG. 1

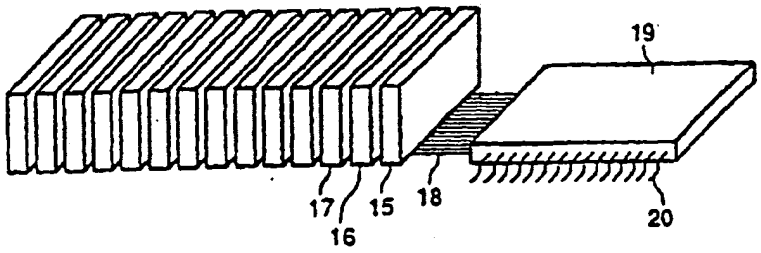
19
17
16
15
18
20

FIG. 3

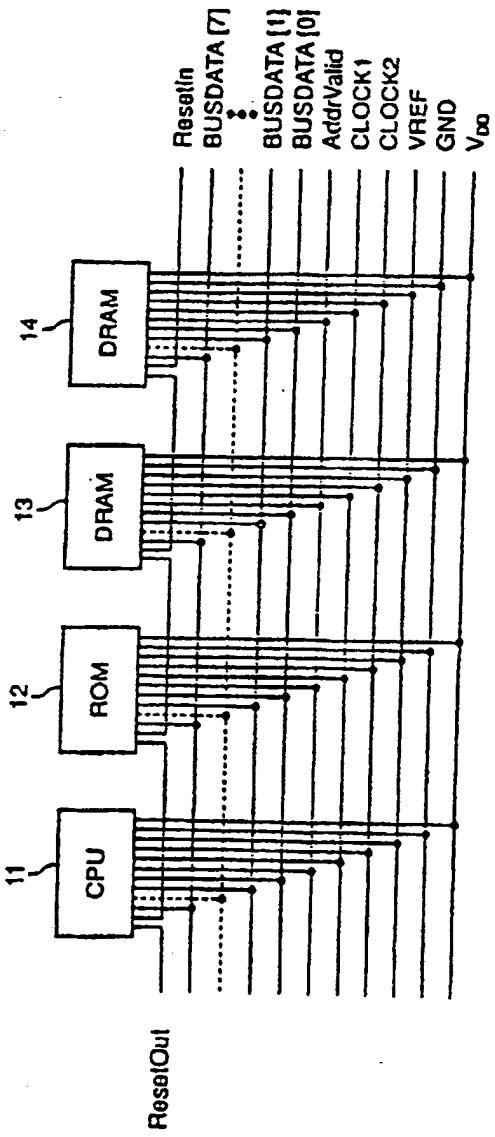
ResetIn
BUSDATA [7]
BUSDATA [1]
BUSDATA [0]
AddrValid
CLOCK1
CLOCK2
VREF
GND
V_{DD}
14
DRAM
13
DRAM
12
ROM
11
CPU
ResetOut

FIG. 2

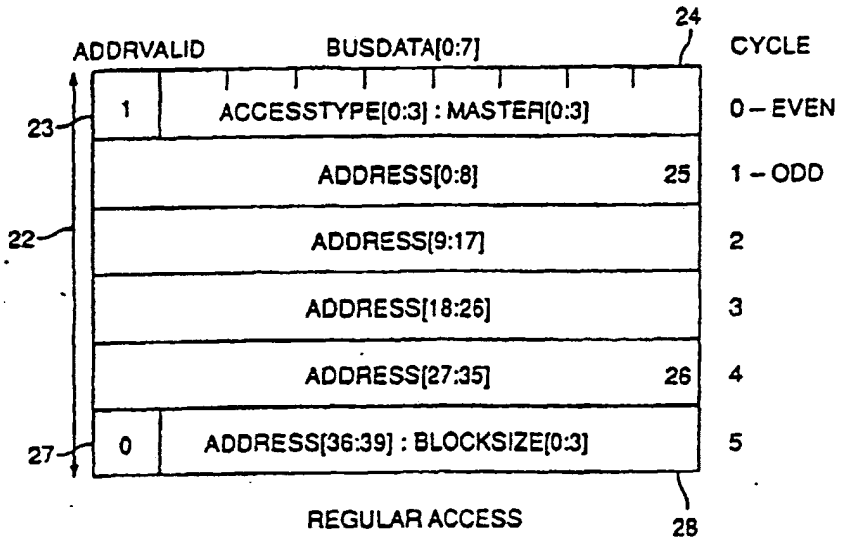
24
ADDRVALID
BUSDATA[0:7]
CYCLE
23
1
ACCESSTYPE[0:3] : MASTER[0:3]
0 – EVEN
ADDRESS[0:8]
25
1 – ODD
22
ADDRESS[9:17]
2
ADDRESS[18:26]
3
ADDRESS[27:35]
26
4
27
0
ADDRESS[36:39] : BLOCKSIZE[0:3]
5
REGULAR ACCESS
28

FIG. 4

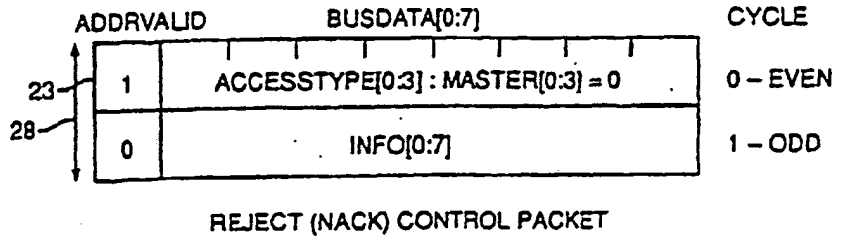
ADDRVALID
BUSDATA[0:7]
CYCLE
23
1
ACCESSTYPE[0:3] : MASTER[0:3] = 0
0 – EVEN
28
0
INFO[0:7]
1 – ODD
REJECT (NACK) CONTROL PACKET

FIG. 5

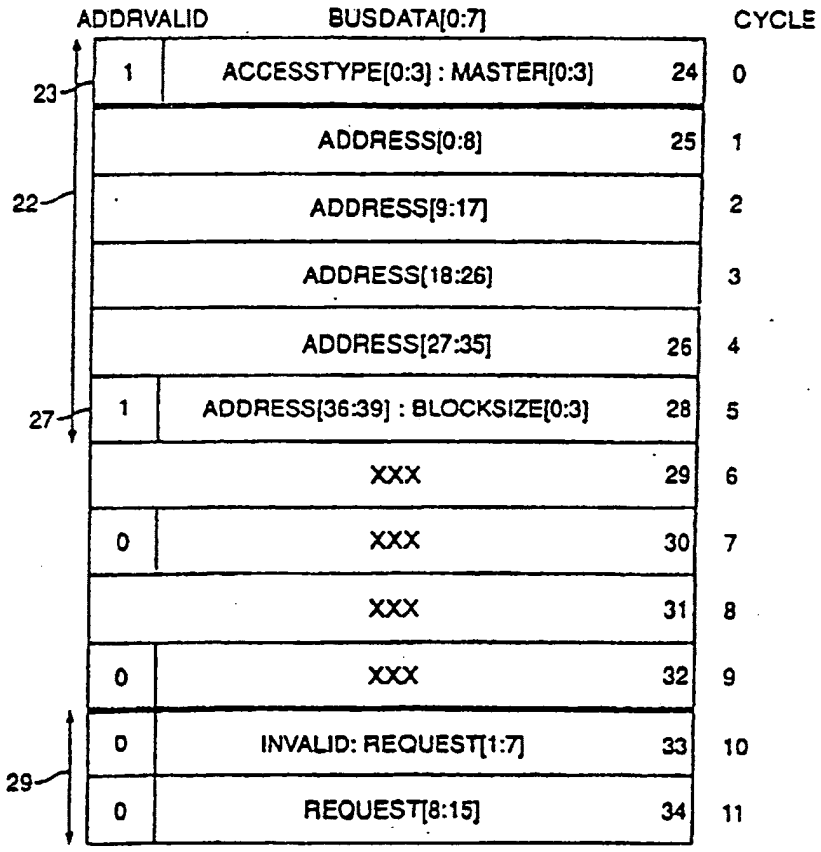
ADDRVALID
BUSDATA[0:7]
CYCLE
23
22
27
29
1
ACCESSTYPE[0:3] : MASTER[0:3]
24
0
ADDRESS[0:8]
25
1
ADDRESS[9:17]
2
ADDRESS[18:26]
3
ADDRESS[27:35]
26
4
1
ADDRESS[36:39] : BLOCKSIZE[0:3]
28
5
XXX
29
6
0
XXX
30
7
XXX
31
8
0
XXX
32
9
0
INVALID: REQUEST[1:7]
33
10
0
REQUEST[8:15]
34
11

FIG. 6


41
42
43
A
B
44
a
b
45

FIG. 7a

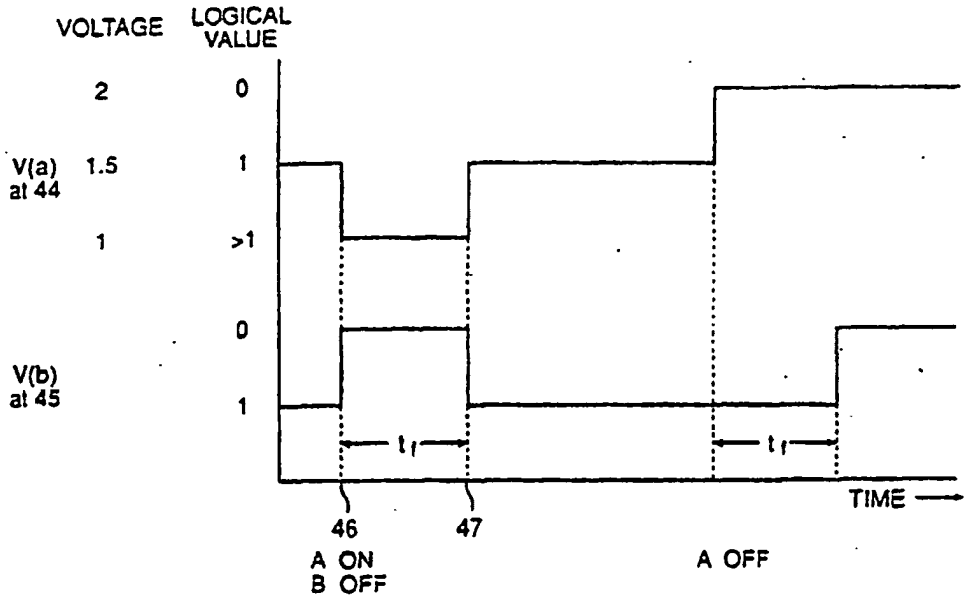
VOLTAGE
LOGICAL VALUE
2
0
V(a) at 44
1.5
1
1
>1
0
V(b) at 45
1
t_f
t_f
TIME
46
47
A ON
B OFF
A OFF

FIG. 7b

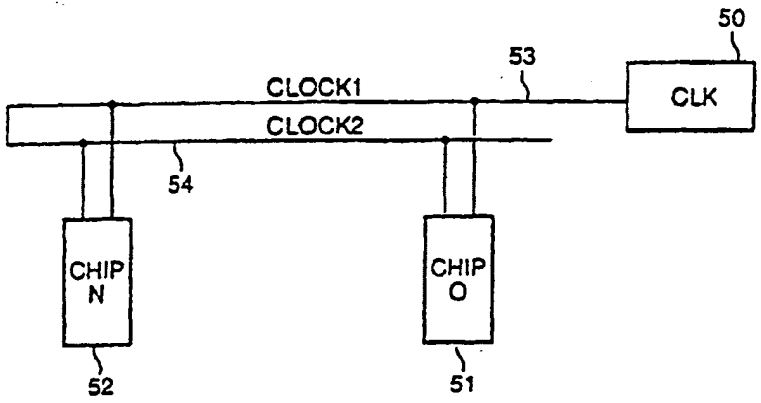
50
53
CLOCK1
CLOCK2
CLK
54
CHIP N
CHIP O
52
51

FIG. 8a

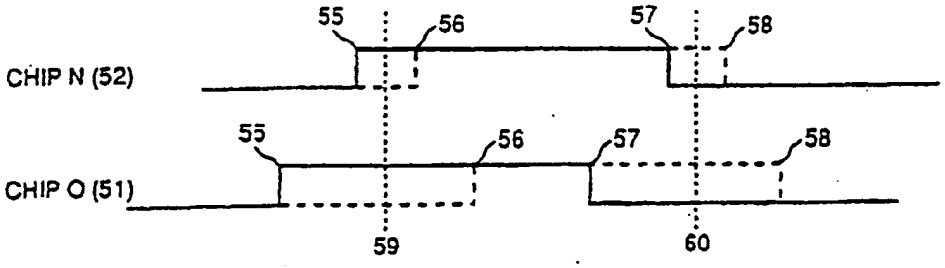
55
56
57
58
CHIP N (52)
55
56
57
58
CHIP O (51)
59
60

FIG. 8b

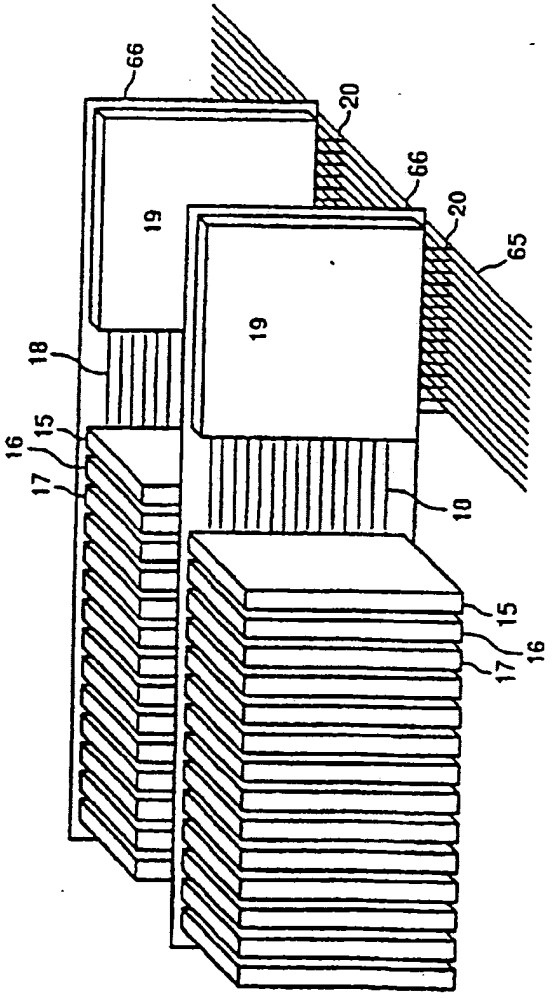

FIG. 9

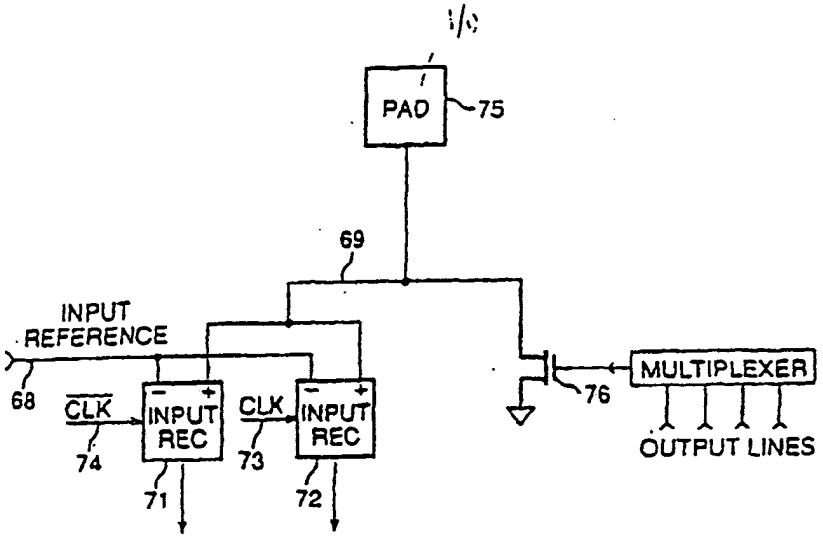
I/O
PAD
75
69
INPUT
REFERENCE
68
CLK
74
INPUT
REC
71
CLK
73
INPUT
REC
72
76
MULTIPLEXER
OUTPUT LINES

FIG. 10

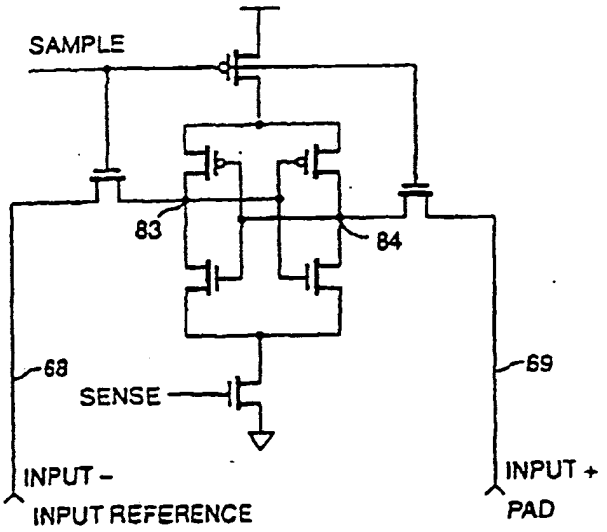
SAMPLE
83
84
68
SENSE
69
INPUT –
INPUT REFERENCE
INPUT +
PAD

FIG. 11

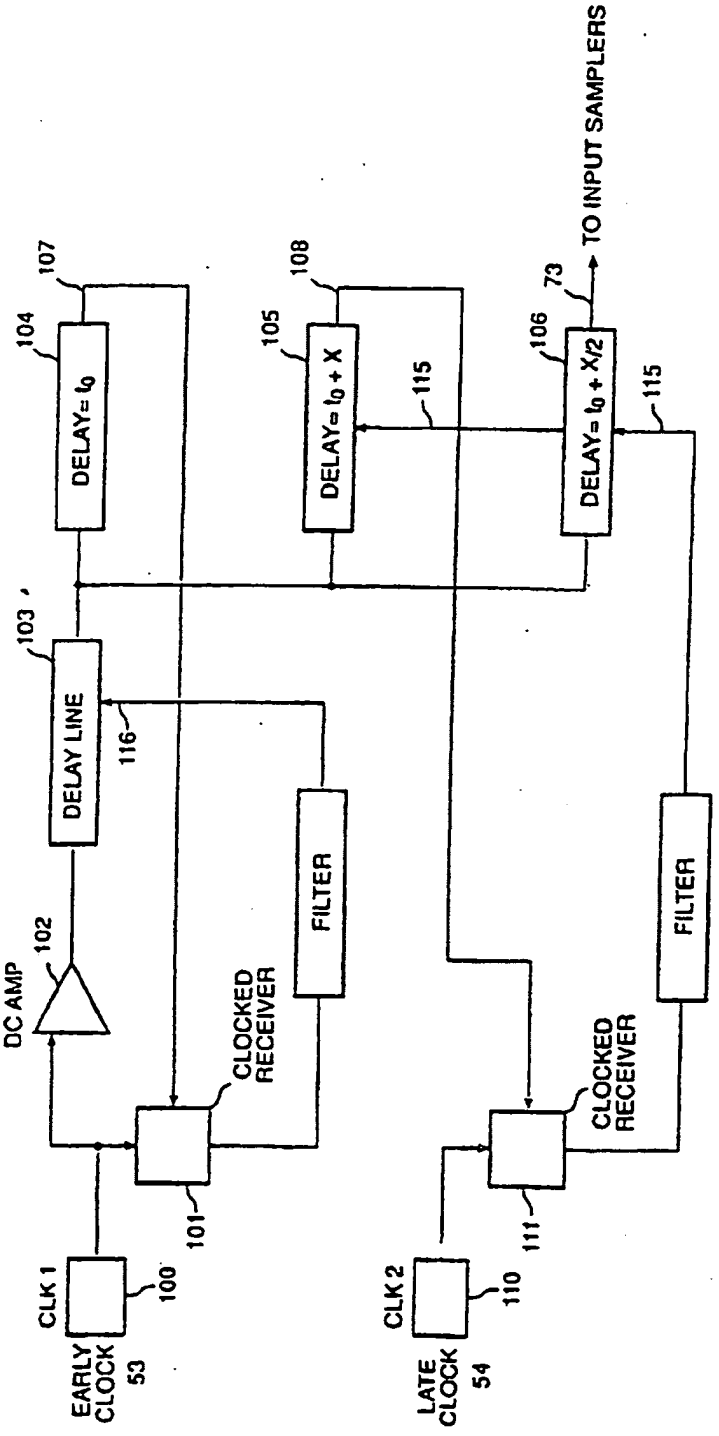
TO INPUT SAMPLERS
73
107
108
104
105
106
DELAY= t_0
DELAY= t_0 + X
DELAY= t_0 + X/2
115
115
103
DELAY LINE
116
DC AMP
102
CLOCKED RECEIVER
FILTER
CLOCKED RECEIVER
FILTER
101
111
CLK 1
100
CLK 2
110
EARLY CLOCK
53
LATE CLOCK
54

FIG. 12

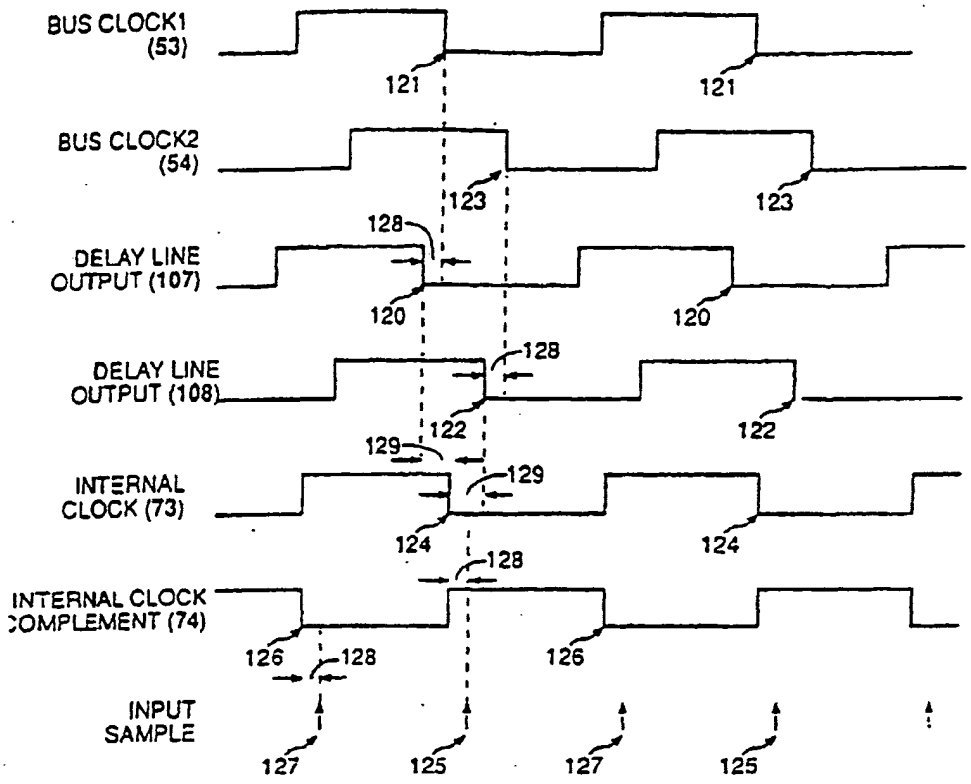
BUS CLOCK1 (53)
121
121
BUS CLOCK2 (54)
123
123
128
DELAY LINE OUTPUT (107)
120
120
128
DELAY LINE OUTPUT (108)
122
122
129
129
INTERNAL CLOCK (73)
124
124
128
INTERNAL CLOCK COMPLEMENT (74)
126
128
126
INPUT SAMPLE
127
125
127
125

FIG. 13

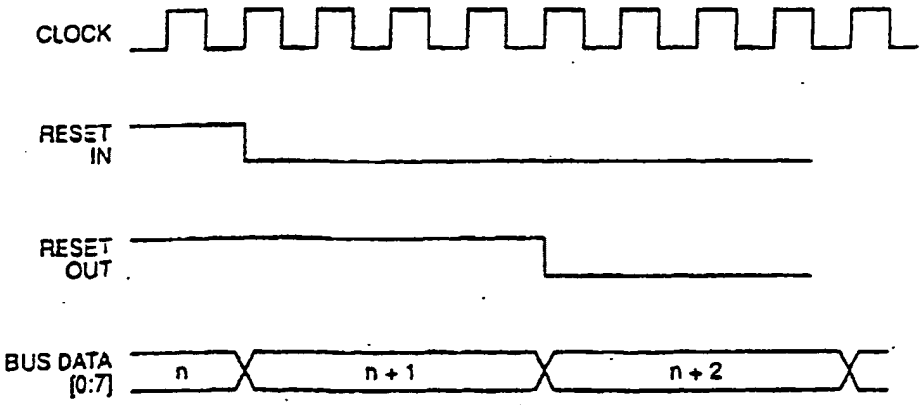
CLOCK
RESET IN
RESET OUT
BUS DATA [0:7]
n
n + 1
n + 2

FIG. 14

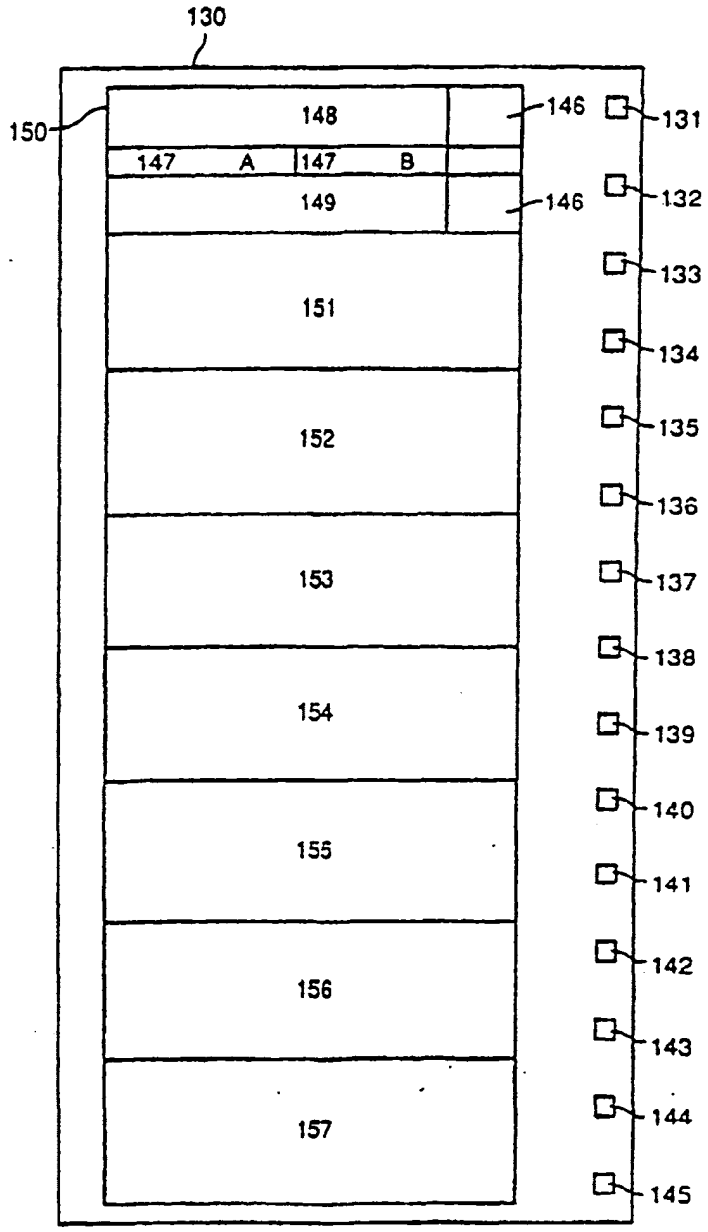
130
150
148
146
131
147
A
147
B
149
146
132
151
133
152
134
153
135
154
136
155
137
156
138
157
139
140
141
142
143
144
145

FIG. 15